TECHNICAL REPORT TR-72-14

TRACKING STUDY: AN INTRODUCTION TO THE USE OF KALMAN FILTERS

by

A L C Quigley

ADMIRALTY SURFACE WEAPONS ESTABLISHMENT
PORTSMOUTH HANTS

UNLIMITED

TECHNICAL REPORT TR-72-14
April 1972

TRACKING STUDY: AN INTRODUCTION TO THE USE OF KALMAN FILTERS

by A L C Quigley

Approved by P R R Clynick
Head of XRP Division

ABSTRACT

It is shown that Kalman filtering may be applied to the radar track-while-scan problem. No attempt is made to rigorously derive the Kalman equations, but the equations are related to more familiar ideas.

It is demonstrated that the least squares $\alpha\beta$ equations constitute a special case of the Kalman filter. The approach used, however, does not require a constant data rate or constant measurement accuracy, meaning that information from various sensors (including links) may be used.

ADMIRALTY SURFACE WEAPONS ESTABLISHMENT
PORTSMOUTH HANTS

LIST OF CONTENTS

LIST OF ILLUSTRATIONS

INTRODUCTION

1. This report is intended to be the first of a series resulting from a study on the general subjects of target tracking and association. The purpose of the study is to produce a generalised package, a subset of which would be chosen for any particular system with any particular combination of sensors. Alternatively, the generalised package could be used as a standard with which to compare simplified processes.

2. The aim of the first phase of the work is to produce a generalised tracking system capable of using elevation and doppler information, as well as having a predictable performance against manoeuvring vehicles. Because this present work does not include any turn handling capabilities, it is not considered a viable process on its own, although some people have used such a method on its own with limited success in other applications.

3. Many of the ideas presented may be expressed in a considerably simpler way by using some matrix theorems. This has been avoided since the author assumes very little knowledge of matrix algebra on the part of the reader; one purpose of the report being to indicate the benefits of this approach.

4. This report is thus concerned with deriving equations of motion of tracks from information from imperfect sensors, and minimising the noise on these tracks. This report considers only two dimensional applications.

5. It is shown how information from a one-dimensional sensor may be incorporated.

WHY USE A COMPUTER TO FORM TRACKS FROM SURVEILLANCE INFORMATION?

6. We need to remind ourselves of the answers to such fundamental questions in order to keep sight of our objectives. The author sees three basic reasons.

a. To provide a tactical picture

A display of the tracks of all vehicles observed by sensors showing present positions, courses and speeds etc is essential for the deployment of a ship and its weapons. The computer enables rapid use to be made of sensor information thereby ensuring that the picture is accurate and up to data.

b. For use in processes such as threat evaluation and weapon assignment.

The computer can forecast future likely positions of tracked vehicles and rapidly perform necessary calculations to assist operators in the assessment of threats and in the optimum weapon deployment to deal with them.

c. To provide target information for weapon deployment.

The computer can be used to generate smooth tracks from noisy information thereby being able to pass information to a weapon sensor more accurately than information derived from a single plot on a display. The computer may also provide continuous information obtained by continually increasing the extent of the extrapolation from its previous best estimate of target position.

The third reason is generally thought to specify the most stringent requirement. However, association processes usually demand the most of the tracking.

7. Thus the uses of the computer are basically to provide quick calculations using input data and to remove the tedium from man's task in tracking, ie taking

care of most tracks (and possibly new detections), while the man would be able to concentrate on the overall situation. The man would also need to handle those situations at which the computer is less capable; such situations would generally be ones which require the use of qualitative information for their solutions.

SUMMARY OF PRESENT METHODS

8. a. The traditional method (known as α-β tracking) is well documented (for example see Reference 1-4).

The method is basically as follows: a forecast position of the vehicle being tracked is calculated, and a fraction of the difference between this forecast and the observed positions is added to the forecast to give an estimate of true position at the time of the observation. The assumed velocity is incremented by a fraction of the difference between observed and forecast positions, divided by the time between observations. The next forecast position is then the present estimate of position extrapolated using the estimated velocity. The process is started by calculating a velocity from the first two observations, and using this to forecast from the second observed position.

The equations are thus:

$$G_n = F_n + \alpha (P_n - F_n)$$

$$V_n = V_{n-1} + \frac{\beta}{T} (P_n - F_n)$$

$$F_{n+1} = G_n + V_n T$$

where:

F_n	=	forecast position for nth measurement
P_n	=	nth measured position
G_n	=	estimated position after nth measurement
V_n	=	estimate of velocity after nth measurement
T	=	time interval between measurements
α	=	position damping factor
β	=	velocity damping factor

b. In most previous real time weapons systems, these equations have been applied in a Cartesian co-ordinate system, one set of equations being used for each dimension. It may be seen that, for positional smoothing, if $\alpha = 0$, all sensor information is ignored whereas if $\alpha = 1$, there is no smoothing of positional information. Similarly $\beta = 0$ causes sensor information to be ignored in the estimation of velocity whereas $\beta > 1$ will cause overcorrection (ie noise amplification, with the ability to predict inside turns). Simpson (Reference 1) calculated the limits of stability in terms of α and β, the result of which is that α and β should normally be between 0 and 1.

The simplest possible system is one in which α and β are fixed constants, and the theory of such systems has been studied by Simpson. For such systems, Bordner and Benedict showed that, for a manoeuvre expressible as a ramp input of position, the relationship between α and β which optimises noise response for a given manoeuvre and also manoeuvre response for a given noise is:-

$$\beta = \frac{\alpha^2}{2-\alpha}$$

c. Constant parameter systems suffer from the incompatible demands that good smoothing requires heavy damping (ie small values of α and β), while good response to manoeuvres requires light damping (ie α and β large). Light damping and therefore poor noise response, can lead to low probabilities of weapon sensor acquisition and plot-to-track association problems. Heavy damping, implying poor manoeuvre response, can cause sudden loss of tracks (sometimes termed track death) through failure to associate with subsequent plots.

d. These limitations led some workers to opt for variable parameter systems where α and β are varied according to the state of the track. Some systems have been developed wherein α and β were initially selected arbitrarily, and changed during program development by trial and error, various operational sets of values being derived for various states of track. Such methods are adaptive and are usually economical in computer use, both in terms of required storage and run time, but generally have no theoretically optimum adaptation.

e. More recently, processes have been developed in which α and β are made to change with time in order to continually compute the least squares line through the observations. Such approaches assumed that errors are equally distributed in x and y and had a constant standard deviation. The formulae for changing α and β in this manner were worked out by Marks (Reference 2). For the incorporation of the nth measurement:-

$$\alpha = \frac{2(2n-1)}{n(n+1)} \tag{1}$$

$$\beta = \frac{6}{n(n+1)} \tag{2}$$

f. It is clear that, for large n, α and β tend to 0, ie observations will be increasingly ignored. This suggests that there should be some maximum value of n. To the best of the author's knowledge, the maximum value used is generally 7 to 15. Such a method may be made adaptive if a means of detecting changes in motion is used, ie if turn detection is provided. Then, if a turn is detected, the values of α and β may be raised simply by lowering n. Doing this will improve the turn following capability. Another approach to turn following is to assume the turn is circular, and to try to track round this. However, this requires that the turn be quickly detected and that turns generally be of sufficiently long duration.

g. As has been previously stated, the preceding methods are fundamentally Cartesian. There are also methods in which the track data is held as x-, and y- co-ordinates, course and speed. Assuming that, for an aircraft the course is more likely to change than the speed, this method allows speed to be more heavily damped than course.

h. Magowan (Reference 3) took a fresh look at the subject and produced a logic which takes account of the usually polar nature of plot noise. This automatically allows for the fact that the area of uncertainty round a plot is a function of range as well as the range and bearing errors. The method stored track data in range, bearing, course and speed. However, the main difficulty of the approach is that the last best estimate of position of a track is treated as having zero error.

k. More recently still, Clynick and Milner (Reference 4) proposed to calculate an optimum time constant (ie, in effect an optimum value of n) based on the target range and derived velocity. This was an attempt to achieve a reasonable compromise performance in calculating course and speed over a considerable range of operating conditions.

m. The approach followed in this report is based on Kalman filter theory, which can take the polar nature of plot noise into account and can provide the exponential weighting in the form proposed in Reference 4. It is not proposed to derive the Kalman filter equations in this report, since this has been done adequately in References 5, 6, 7 and 8. Instead, a "picture" will be given for a single dimension case.

A SIMPLE APPROACH TO LEAST SQUARES SMOOTHING

9. Before dealing with the Kalman filter, we shall try to introduce the concepts involved in smoothing, and to show why a least squares criterion is used.

a. Let us assume that we have two independent estimates, x'_2 and x_2 of the same variable, with mean square errors (variances) a' and r respectively. Assuming that the errors have Gaussian distributions then from the first estimate, the p.d.f. for the probability that the true value is x is:-

$$p_1 = \frac{1}{\sqrt{2\pi a'}} \exp\left\{ - \frac{(x'_2-x)^2}{2a'} \right\}$$

Similarly, from the second estimate, the p.d.f. for the probability that the true value is x is:

$$p_2 = \frac{1}{\sqrt{2\pi r}} \exp\left\{ - \frac{(x_2-x)^2}{2r} \right\}$$

The joint probability that the true value, from both estimates, is x is $p = p_1 p_2$ where

$$p = \frac{1}{2\pi\sqrt{a'r}} \exp\left\{ - \frac{(x'_2-x)^2}{2a'} - \frac{(x_2-x)^2}{2r} \right\}$$

Now it may be easily shown (see Appendix C) that the product of these two Gaussian distributions is another Gaussian distribution given by

$$p = \text{constant x} \exp\left\{ - \frac{(x-\hat{x})^2}{2\hat{\sigma}^2} \right\}$$

where $\hat{x} = (x'_2\, r + x_2 a')/(a'+r)$ (3)

and $\hat{\sigma}^2 = a'r/(a'+r)$ (4)

This joint probability is therefore maximised when $x = \hat{x}$, while the standard deviation associated with $\hat{x}$ is $\hat{\sigma}$, which is always less than a' and $\sqrt{r}$. The actual value of the constant in the expression for p does not matter here, since we do not want to know the actual value of ρ, but rather the value of x which maximises it. (The "constant" in the expression is not strictly a constant, since it is a function of the a' and r, x'_2 and x_2, but the important point is that it is not a function of x which is the quantity of interest.)

b. It is no doubt apparent that x'_2 could have represented a forecast of x, and that x_2 could have represented a measurement. If x changes with time, it is also necessary to estimate the rate of change (which we will assume to be constant).

c. Let us consider the fact that x'_2 is an estimate based on a previous (estimated) position $\hat{x}_1$ and there is an estimated velocity $\hat{v}_1$, and that x_2 is an observation. Let us assume that x'_1 has variance a' (as before) and that x_2 has variance r. We also assume that $\hat{x}_1$ has variance a, that $\hat{v}_1$ has variance d, and that the covariance between $\hat{x}_1$ and $\hat{v}_1$ is b. (The covariance of $\hat{x}_1$ and $\hat{v}_1$ is defined thus: if an error in $\hat{x}_1$ is $\delta\hat{x}_1$ and in $\hat{v}_1$ is $\delta\hat{v}_1$ then the covariance of $\hat{x}_1$ and $\hat{v}_1$ is the mean value of all products $(\delta\hat{x}_1\ \delta\hat{v}_1)$. This is analogous to the variance of a single variable being the mean square error: the covariance is the mean value of the product of errors in two variables.)

Appendix C shows that the best estimate of velocity after making the measurement x_2 may be written:-

$$v = \hat{v}_1 + \frac{(b+td)(x_2-x'_2)}{a+2bt+t^2d+r} \qquad (5)$$

where v_1 = previous estimate of velocity

b = covariance of previous position and velocity estimates

a = variance of previous position estimate

t = measurement time interval

d = variance of previous velocity estimate

and that its variance is given by s^2 where:

$$s^2 = \frac{d(a+r)-b^2}{a+2bt+t^2d+r} \qquad (6)$$

We may reconfigure (3) to have the same form as (5) hence:-

$$\hat{x}_2 = x'_2 + \frac{a'(x_2-x'_2)}{a'+r} \qquad (7)$$

or, in terms of the a, b, d etc,

$$\hat{x}_2 = x'_2 + \frac{(a+2bt+t^2d)(x_2-x'_2)}{a+2bt+t^2d+r} \qquad (8)$$

Hence we may write:-

$$\hat{x}_2 = x'_2+\alpha(x_2-x'_2) \qquad (9)$$

$$\hat{v}_2 = \hat{v}_1+\frac{\beta}{t}(x_2-x'_2) \qquad (10)$$

where $\alpha = \frac{a'}{a'+r} = \frac{a+2tb+t^2d}{a+2tb+t^2d+r}$

and $\beta = \frac{t(b+td)}{a+2tb+t^2d+r}$

Now, $\hat{\sigma}^2$ is the new value of a, and s^2 is the new value of d. It can readily be shown that the new value of b is

$$\frac{(b+td)r}{a+2bt+t^2d+r} .$$

d. Initial Conditions

Let us consider two initial measurements x_0 and x_1 taken time t apart. The best estimate of position is thus x_1, and the estimate of velocity is

$$\frac{x_1-x_0}{t} .$$

If there is an error in measurement of x_0 of δx_0 and one in x_1 of δx_1, then the error in estimating velocity is $(\delta x_1 - \delta x_0)/t$.

The variance of this velocity estimate is the mean value of

$$\frac{(\delta x_1 - \delta x_0)^2}{t^2} = \frac{(\delta x_1)^2 - 2\delta x_0 \delta x_1 + (\delta x_0)^2}{t^2} .$$

The mean value of $\delta x_0 \delta x_1 = 0$ for uncorrelated measurements, hence variance of $v_1 = d = (\text{mean } (\delta x_1)^2 + \text{mean } (\delta x_0)^2)/t^2$

$= (r_1 + r_0)/t^2$, say.

The covariance of $\hat{x}_1$ and v_1 (called b) is the mean value of:

$$\frac{\delta x_1(\delta x_1 - \delta x_0)}{t}$$

which gives $b = \frac{r_1}{t}$.

Thus we may construct a covariance matrix which relates to $\hat{x}_1$ and v_1 and is:

$\begin{bmatrix} a & b \\ b & d \end{bmatrix}$ whose elements are the mean values of the elements of

$$\left(\begin{bmatrix} \delta \hat{x}_1 \\ \delta v_1 \end{bmatrix} [\delta \hat{x}_1 \quad \delta v_1] \right)$$

where

$$a = r_1,\ b = \frac{r_1}{t} \text{ and } d = \frac{r_1+r_0}{t^2} .$$

We would have obtained the same result by applying Appendix B equation 3 to:

$$\begin{bmatrix} r_0 & 0 \\ 0 & r_1 \end{bmatrix}, \text{ using the transformation } \begin{bmatrix} x_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{1}{t} & \frac{1}{t} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}.$$

We wish to forecast over a time interval t to calculate a forecast position x'_2.

We thus use the transformation

$$\begin{bmatrix} x'_2 \\ v_1 \end{bmatrix} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ v_1 \end{bmatrix} \tag{11}$$

We may apply Appendix B equation 3 to calculate the covariance matrix for the forecast state

$$\begin{bmatrix} x'_2 \\ v_1 \end{bmatrix}. \text{ This is}$$

$$\begin{bmatrix} a+2bt+t^2d & b+td \\ b+td & d \end{bmatrix} \tag{12}$$

If the variances of measurement error are assumed equal and since

$a = r_1$, $b = \frac{r_1}{t}$, $d = \frac{r_1+r_0}{t^2}$ and writing $\sigma^2 = r_1 = r_0$

we find that the covariance matrix for

$$\begin{bmatrix} x'_2 \\ v_1 \end{bmatrix} \text{ is: } \begin{bmatrix} 5\sigma^2 & \frac{3\sigma^2}{t} \\ \frac{3\sigma^2}{t} & \frac{2\sigma^2}{t^2} \end{bmatrix}$$

When we combine the forecast x'_2 with the observation x_2, which has variance r using (5) and (8) we get

$$v = \hat{v}_1 + \frac{3\sigma^2(x_2-x'_2)}{t(5\sigma^2+\sigma^2)}$$

$$= \hat{v}_1 + \frac{(x_2-x'_2)}{2t}$$

$$\hat{x}_2 = x'_2 + \frac{5\sigma^2(x_2-x'_2)}{5\sigma^2+\sigma^2}$$

$$= x'_2 + \frac{5(x_2-x'_2)}{6}$$

Thus, for constant variances of observation of errors, we find, using the definitions of the two damping factors given by (9) and (10), that $\alpha = 5/6$ and $\beta = 1/2$ are the optimum values for incorporation of the third plot into the track.

Marks (reference 2), has shown that, for constant errors and time intervals:-

$$\alpha = \frac{2(2n-1)}{n(n+1)}$$

$$\beta = \frac{6}{n(n+1)}$$

where n is the observation number, the first observation for the track having $n = 1$.

In the above example $n = 3$, which gives $\alpha = 5/6$ and $\beta = 1/2$.

Now in (9), α is effectively defined as the ratio:

$$\frac{\text{variance of forecast position}}{\text{variance of forecast position + variance of observation}}$$

If α is less than 0.5 it therefore means that the variance of forecast position is less than the variance of the observed position. When $n = 6$, $\alpha = 11/21$; when $n = 7$, $\alpha \approx 13/28$. Thus, when $n = 7$ the forecast is about as accurate as the measurements. Also, the variance of the smoothed position = αr, which, for $n = 7$ gives $13r/28$. Thus when $n = 7$ the variance of the smoothed position is just less than half the measurement variance, ie the "noise power" has been halved.

It is not proposed to go any further with this example, since its purpose was solely to introduce the basic concepts of tracking, but the illustration serves to demonstrate the following drawbacks of this simple approach.

c. The example was given for tracking in one dimension only, and assumed that the time interval between observations was constant. If we wish to track in more than one dimension, this simple theory can only work if the measurement errors in all the dimensions are not related in any way (ie are uncorrelated), in order that the tracking may be separately performed in those dimensions. Now, radars with auto-extraction equipment have independent errors in range and bearing. This would constrain tracking to tracking in range and bearing: linear forecasting is then not good enough for finite range and bearing rates. The theory is only valid for tracking in x and y when the variances in x and y can be considered independent and constant. This is so in manual detection systems which employ Cartesian frames of reference.

It should be remembered that, if variances in range and bearing are constant, the variances of x and y after transformation from the r&θ to the x&y co-ordinate system will be related and will be functions of r and θ (or x and y) and will not, therefore, be constant for a moving target.

The Kalman filter, which forms the basis of this tracking study, does not suffer from the restrictions above: it is a generalised version of the method empirically derived and is in matrix form - thereby conveniently handling the correlations between errors. The Kalman filter equations were first derived by Kalman and Bucy, (References 6, 7), but many others have produced different derivations from different viewpoints (for example see References 5 and 8).

10. KALMAN FILTERING

a. In this section, all upper case letters will refer to matrices and lower case letters to scalars. We will take the ideas already presented and represent them in a different way as an introduction to the Kalman filter equations.

Equation (11) states:-

$$\begin{bmatrix} x'_2 \\ v_1 \end{bmatrix} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ v_1 \end{bmatrix}$$

which means that the state of the track which is given by $\hat{x}_1$ and v_1 is transformed to another state given by x'_2, v_1 over a time interval t. We may represent

$\begin{bmatrix} \hat{x}_1 \\ v_1 \end{bmatrix}$ as $\hat{X}_1$ and hence represent $\begin{bmatrix} x'_2 \\ v_1 \end{bmatrix}$ as X'_2.

The matrix $\begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}$ which we may call Φ transforms $\hat{X}_1$ to X'_2.

We may thus write equation (11) as $X'_2 = \Phi\hat{X}_1$ or, in terms of transition from state $\hat{X}_k$ to state X'_{k+1},

$$X'_{k+1} = \Phi\hat{X}_k \ . \tag{13}$$

This expression is in effect the equation of motion from which a future state X'_{k+1} can be calculated from a past state $\hat{X}_k$.

The covariance matrix associated with $\hat{X}_1$ (or $\hat{X}_k$) was given in (5) as

$$\hat{P}_1 = \begin{bmatrix} a & b \\ b & d \end{bmatrix} \quad (= \hat{P}_k).$$

That associated with X'_2 (or X'_{k+1}) was found in Appendix C as

$$P'_2 = \begin{bmatrix} a+2bt+t^2d & b+td \\ b+td & d \end{bmatrix} \quad (= P'_{k+1}).$$

In fact the covariance matrices are related by the equation:

$$P'_{k+1} = \Phi \hat{P}_k \Phi^T,$$

the superscript ^{T} indicating the matrix transpose. This may be checked by deriving $\Phi P_1 \Phi^T$ and comparing with P'_2.

α was defined by (9) as: $\frac{a+2bt+t^2d}{a+2bt+t^2d+r}$, r being the measurement variance

and β was defined by (10) as: $\frac{(b+td)t}{a+2bt+t^2d+r}$.

The denominator of both of these quantities is the (1,1) element of P'_2 plus r. We may obtain this (1,1) element of P'_2 by the operation

$$[1 \quad 0]\, P'_2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} = a+2bt+t^2d.$$

Hence, if we define a matrix $M = [1\ 0]$, the denominator for α and β is (MPM^T+R), where R is the measurement covariance matrix (which is the same size as the MPM^T product and is a 1X1 matrix in our example).

The numerators for α and β/t are the contents of the first column of P'_{k+1} in this example, and we may obtain these thus:

$$P'_{k+1}\, M^T = \begin{bmatrix} a+2bt+t^2d \\ b+td \end{bmatrix}$$

Thus $P'_{k+1}M^T\,(MPM^T+R)^{-1}$ gives a matrix which we shall call K_{k+1}

where the first element of K_{k+1} is α and the second is $\frac{\beta}{t}$.

Our smoothing equations are:

$$\hat{x}_{k+1} = x'_{k+1} + \alpha(x_{k+1} - x'_{k+1})$$

$$v_{k+1} = v_k + \frac{\beta}{t}\,(x_{k+1} - x'_{k+1})$$

which is clearly the same as writing:

$$\hat{X}_{k+1} = X'_{k+1} + K_{k+1}\,(Y_{k+1} - MX'_{k+1})$$

where Y_{k+1} is the matrix of observations, and is 1X1 for this case.

It is more usual to write this as:

$$\hat{X}_{k+1} = X'_{k+1} - K_{k+1}\,(MX'_{k+1} - Y_{k+1}).$$

The variance of $\hat{x}_{k+1}$ was found in (4) to be: $\theta^2 = \frac{ra'}{r+a'}$.

In (12), a', the variance of forecast position, was found to be given by:

$$a' = a+2bt+t^2d.$$

Hence we may rewrite (4):

$$\theta^2 = \frac{r(a+2bt+t^2d)}{a+2bt+t^2d+r}.$$

We may further rewrite this as follows:

$$\hat{\sigma}^2 = a+2bt+t^2d - \frac{(a+2bt+t^2d)^2}{a+2bt+t^2d+r}$$

Now, $MP'_{k+1} = [a+2bt+t^2d \quad b+td]$

$\therefore$ The (1,1) element of $K_{k+1}MP'_{k+1} = \dfrac{(a+2bt+t^2d)^2}{a+2bt+t^2d+r}$

Thus it is correct, for the (1,1) term at least, to write:

$$\hat{P}_{k+1} = P'_{k+1} - K_{k+1}MP'_{k+1}$$

We may now move on to a more formal definition of the Kalman filter.

b. The Kalman filter equations which are derived in References 6-8 may be written thus:

$$X'_{k+1} = \Phi_k \hat{X}_k \tag{14}$$

$$P'_{k+1} = \Phi_k P_k \Phi_k^T + G_k Q_k G_k^T \tag{15}$$

$$K_{k+1} = P'_{k+1} M^T_{k+1} \left[M_{k+1} P'_{k+1} M^T_{k+1} + R_{k+1}\right]^{-1} \tag{16}$$

$$\hat{X}_{k+1} = X'_{k+1} - K_{k+1} (M_{k+1} X'_{k+1} - Y_{k+1}) \tag{17}$$

$$P_{k+1} = P'_{k+1} - K_{k+1} M_{k+1} P'_{k+1} \tag{18}$$

where a prime (') represents a forecast value, a hat (^) represents a best estimate of a variable, superscript ^{T} is the transpose of a matrix and:

X is the state vector of order nx1

Φ is thetransition matrix, nxn, and defines the transition from true state X_k to true state X_{k+1}

P is the nxn covariance matrix of the estimate of X (ie P'_{k+1} is the covariance matrix for estimate X'_{k+1})

G is nxj matrix representing the effects of j elements of 'plant noise' sometimes known as process noise on the n elements of the state vector;

Q is a jxj matrix, being the covariance matrix of the plant noise

Y is an mx1 vector and has as its elements the measured variables

M is the (so-called) measurement matrix and is mxn

R is the mxm covariance matrix of the measurements Y

K is a matrix, nxm and essentially contains the damping factors.

c. Some explanation of these terms

The true state at time t_k is X_k. The true state at time t_{k+1} is

$$X_{k+1} = \Phi_k X_k + G_k U_k$$

where U_k is a set of random inputs known as plant noise. In essence, the G, U, Q terms represent how true the equations of motion implied in Φ are, or, for example, how straight a nominally straight track really is. This noise is assumed to have zero mean, thus the best forecast is as given in 13. Having thus mentioned plant noise, we will now ignore it for the purposes of the rest of the report, since the main effect is analogous to the time constant concept in Reference 4 which is not of concern at present, but will be discussed in a future report. (It may be apparent to the reader that the effects of plant noise reduce the rates at which variances of successive estimates change.) Other terms may be used in Kalman filters, these are also ignored for the purposes of the report.

d. Incorporating a measurement

When a measurement is made, what is actually observed is some function of the true state together with noise.

Thus what is observed is:

$$Y_{k+1} = M_{k+1} X_{k+1} + N_{k+1} \tag{19}$$

where X_{k+1} is true state at $t = t_{k+1}$, and N_{k+1} is a set of noise components. This equation then defines M_{k+1}.

e. Working through the Kalman Equations

As a simple example consider a state vector to represent a position (say x) and a velocity (say $\dot{x}$). Let us consider that x is observable (as Y = x+n) but $\dot{x}$ is not. Then:

X represents $\begin{bmatrix} x \\ \dot{x} \end{bmatrix}$ and Y represents a measurement of x, thus Y will be a 1x1 matrix which is the true value of x corrupted by noise. In this example M is obviously a 1x2 matrix:

$$[1 \quad 0]$$

Hence $Y = [1 \quad 0] \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + [N]$

f. Forecasting

In this simple example we assume linear equations of motion, hence:

$$x' = x + \dot{x}(\Delta t) \qquad (\Delta t \text{ being } t_{k+1}-t_k)$$

$$\dot{x}' = \dot{x}$$

or, in matrix form

$$\begin{bmatrix} x' \\ \dot{x}' \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

Comparison with (14) shows that we have defined Φ to be:

$$\begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

To simplify matters slightly, we will write t instead of Δt. This new variable should strictly be written $t_{k,k+1}$ but is written as t for obvious reasons.

Let us assume that, at $t = t_k$, P_k is:

$$\begin{bmatrix} a & b \\ b & d \end{bmatrix}$$

(this matrix is symmetrical since the covariance of x and $\dot{x}$ equals the covariance of $\dot{x}$ and x).

Hence P'_{k+1} =

$$\begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a & b \\ b & d \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t & 1 \end{bmatrix}$$

$$= \begin{bmatrix} a+2bt+t^2d & b+td \\ b+td & d \end{bmatrix}$$

Now K is nxm and, as stated earlier R_{k+1} has one element which we will call r_{k+1}.

$$\therefore \quad M_{k+1} \, P'_{k+1} \, M^T_{k+1} = [1 \quad 0] \begin{bmatrix} a+2bt+b^2d & b+td \\ b+td & d \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= a+2bt+t^2d.$$

$$\therefore \quad K_{k+1} = \begin{bmatrix} a+2bt+t^2d & b+td \\ b+td & d \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} [a+2bt+t^2d+r_{k+1}]^{-1}$$

$$= \begin{bmatrix} \dfrac{a+2tb+t^2d}{a+2tb+t^2d+r_{k+1}} \\[2ex] \dfrac{b+td}{a+2tb+t^2d+r_{k+1}} \end{bmatrix}$$

Now, the expression $a+t(2b+td)$ is the variance of the forecast position x'_{k+1}. If we call this a'_{k+1} then the first element of K_{k+1} is

$$\frac{a'_{k+1}}{a'_{k+1}+r_{k+1}}$$

If this is compared with (9) it will be seen that this is precisely α. It will also be seen that the estimate of velocity variance in (6) was in fact a combination of velocity variance and position velocity covariance. It will then be clear that the second element of K_{k+1} is the $\frac{\beta}{t}$ defined.

g. Smoothing

We can now move on to (17).

$M_{k+1}\ X'_{k+1}$ is simply x'_{k+1}. Thus,

$$\hat{X}_{k+1} = \begin{bmatrix} x'_{k+1} \\ \dot{x}'_{k+1} \end{bmatrix} - \begin{bmatrix} \dfrac{(a+2bt+t^2d)(x'_{k+1} - y_{k+1})}{a+2bt+t^2d+r_{k+1}} \\ \dfrac{(b+td)(x'_{k+1} - y_{k+1})}{a+2bt+t^2d+r_{k+1}} \end{bmatrix}$$

Similarly, using (18),

$$\hat{P}_{k+1} = \begin{bmatrix} a+2bt+t^2d & b+td \\ b+td & d \end{bmatrix} - \begin{bmatrix} \dfrac{a+2bt+t^2d}{a+2bt+t^2d+r_{k+1}} \\ \dfrac{b+td}{a+2bt+t^2d+r_{k+1}} \end{bmatrix} [1\ 0] \begin{bmatrix} a+2bt+t^2d & b+td \\ b+td & d \end{bmatrix}$$

To simplify the manipulation, we write

$a' = a+t(2b+td)$, and $r = r_{k+1}$

then

$$\hat{X}_{k+1} = \begin{bmatrix} x'_{k+1} \\ \dot{x}'_{k+1} \end{bmatrix} - \begin{bmatrix} \dfrac{a'}{a'+r}(x'_{k+1} - y_{k+1}) \\ \dfrac{(b+td)}{a'+r}(x'_{k+1} - y_{k+1}) \end{bmatrix}$$

$$P_{k+1} = \begin{bmatrix} a' & b+td \\ b+td & d \end{bmatrix} - \begin{bmatrix} \dfrac{a'}{a'+r} \\ \dfrac{b+td}{\sigma^2+r} \end{bmatrix} [1\ 0] \begin{bmatrix} a' & b+td \\ b+td & d \end{bmatrix}$$

$$= \begin{bmatrix} a' & b+td \\ b+td & d \end{bmatrix} - \begin{bmatrix} \frac{(a')^2}{a'+r} & \frac{a'(b+td)}{a'+r} \\ \frac{a'(b+td)}{a'+r} & \frac{(b+td)^2}{a'+r} \end{bmatrix} .$$

$$= \begin{bmatrix} \frac{a'r}{a'+r} & \frac{(b+td)r}{a'+r} \\ \frac{(b+td)r}{a'+r} & \frac{d(a+r)-b^2}{a'+r} \end{bmatrix} .$$

The variance of estimated position at (k+1) may be seen to be exactly that derived earlier in the single dimension case. Similarly, as mentioned earlier, the first element of K_{k+1} corresponds exactly with the derived α and the second element with $\frac{\beta}{t}$.

h. <u>Initiation of the filter</u>

The next problem to consider is that of starting up the filter. However, the starting of the filter requires knowledge which may not be available. In particular when we have received one observation, we need a velocity estimate and its variance. Let us consider that we have a stream of measurements y_i each of variance r. Thus,

$$\hat{X}_o = \begin{bmatrix} y_o \\ o \end{bmatrix} \quad \text{assuming zero velocity.}$$

$$\text{Thus } X'_1 = \begin{bmatrix} y_o \\ o \end{bmatrix} \quad \text{at time } \Delta t .$$

We know that the variance of y_o was r. We assumed that $\dot{x}$ was zero, thus we are justified in assigning a large variance to this value. In reality, we may assume initial velocity components such that the target is approaching. It should also be possible to use a sensible value for the variance of this estimate since target speeds are limited.

The covariance between $\hat{x}_o$ and $\hat{\dot{x}}_o$ is zero since the errors in them are not related.

$$\text{Hence } P_o = \begin{bmatrix} r & o \\ o & L \end{bmatrix}, \text{ L being a large number.}$$

$$\text{Thus } P'_1 = \begin{bmatrix} r+t^2L & tL \\ tL & L \end{bmatrix}$$

$$\therefore K_1 = P'_1 \begin{bmatrix} 1 \\ o \end{bmatrix} \begin{bmatrix} [1 \quad o]P'_1 \begin{bmatrix} 1 \\ o \end{bmatrix} + r \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} \frac{r+t^2L}{2r+t^2L} \\ \frac{tL}{2r+t^2L} \end{bmatrix}$$

$$\therefore \quad \hat{X}_1 = \begin{bmatrix} y_0 \\ 0 \end{bmatrix} - \begin{bmatrix} \frac{r+t^2L}{2r+t^2L} \\ \frac{tL}{2r+t^2L} \end{bmatrix} [y_0 - y_1] \, .$$

Since L is large (at present, at any rate)

$$\hat{X}_1 = \begin{bmatrix} y_0 - y_0 + y_1 \\ \frac{y_1 - y_0}{t} \end{bmatrix} = \begin{bmatrix} y_1 \\ \frac{y_1 - y_0}{t} \end{bmatrix}$$

which is perfectly sensible.

$$P'_1 - K_1 M_1 P'_1$$

$$= P'_1 - \begin{bmatrix} \frac{r+t^2L}{2r+t^2L} \\ \frac{tL}{2r+t^2L} \end{bmatrix} [1 \quad 0] \begin{bmatrix} r+t^2L & tL \\ tL & L \end{bmatrix}$$

$$= \begin{bmatrix} \frac{(r+t^2L)(2r+t^2L-r-t^2L)}{2r+t^2L} & \frac{tL(2r+t^2L-r-t^2L)}{2r+t^2L} \\ \frac{tL(2r+t^2L-c^2-t^2L)}{2r+t^2L} & \frac{L(2r+t^2L-t^2L)}{2r+t^2L} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{r(r+t^2L)}{2r+t^2L} & \frac{rtL}{2r+t^2L} \\ \frac{rtL}{2r+t^2L} & \frac{2rL}{2r+t^2L} \end{bmatrix}$$

Now, since L is large,

$$\hat{P}_1 = \begin{bmatrix} r & \frac{r}{t} \\ \frac{r}{t} & \frac{2r}{t^2} \end{bmatrix}$$

The variance of the position x_1 is thus r, and that of estimating $\dot{x}_1$ is $\frac{2r}{t^2}$, both of which are reasonable results, and in keeping with those derived empirically in (4).

We can easily show that the covariance terms, $\frac{r}{t}$ tie in with the empirical theory by calculating P'_2 and showing that the variance of x'_2 is $5\sigma^2$.

$$P'_2 = \Phi P_1 \Phi^T$$

$$= \begin{bmatrix} 1 & t \\ o & 1 \end{bmatrix} \begin{bmatrix} r & \frac{r}{t} \\ \frac{r}{t} & \frac{2r}{t^2} \end{bmatrix} \begin{bmatrix} 1 & o \\ t & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 5r & \frac{3r}{t} \\ \frac{3r}{t} & \frac{2r}{t^2} \end{bmatrix} .$$

The filter may also be started by observing the first two plots and calculating initial velocity, variances etc.

j. Equivalence of Kalman and α-β

Thus the Kalman filter has been shown to be equivalent to the least squares αβ tracker provided that:

(1) all measurements have equal variance

(2) data rate is constant

(3) there is no plant noise (ie the track really does follow the assumed equations of motion).

(4) an error in one co-ordinate of the track's position does not affect the other co-ordinate.

Hence the least squares αβ tracker is a restricted Kalman filter.

Restrictions (1) and (2) become important when information is derived from two or more unsynchronised sensors of different accuracies.

Restriction (3) is difficult to deal with in the αβ tracker. Item (4) is perhaps the most significant restriction. If a track is stored as range, range rate, bearing and bearing rate, then the errors between range and bearing will not be related (unless data is input manually from a device driven in Cartesian manner). If the track is represented in Cartesian co-ordinates and their rates, the errors are related unless data is input from a Cartesian device.

It is shown in Appendix A that the Kalman filter, the least squares αβ tracker and linear regression give the same results, for special conditions mentioned above. When plant noise is absent, the Kalman filter represents multi-dimensional linear regression worked out on a continuous basis.

k. Choice of equation of motion

As has been mentioned previously, radar information is available as a range and bearing, with the errors in the two measurements unrelated. At first sight, it may appear reasonable to attempt to store tracks as range, range rate, bearing and bearing rate, ie to use:

$$\begin{bmatrix} r'_{k+1} \\ \dot{r}'_{k+1} \\ \theta'_{k+1} \\ \dot{\theta}'_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & t & o & o \\ o & 1 & o & o \\ o & o & 1 & t \\ o & o & o & 1 \end{bmatrix} \begin{bmatrix} \hat{r}_k \\ \dot{r}_k \\ \hat{\theta}_k \\ \dot{\theta}_k \end{bmatrix}$$

as $X'_{k+1} = \Phi\hat{X}_k$.

Such a representation attempts to fit a spiral through all plots which is not generally valid particularly for targets which pass by the origin. However, this error is not significant for crossing at zero or large ranges. Crossing at zero range, is of course, radial. If a system is only interested in point defence, such a method may be adequate. The effect may be reduced by use of the time constant giving exponential decay referred to earlier. In the simple $\alpha\beta$ system this would be derived (approximately) by fixing a maximum value of n. In the Kalman filter, introduction of plant noise terms has a similar effect. Such a term must be used anyway, otherwise the filter will eventually ignore plots since the variances of plots would become very large compared with forecast variances.

One would expect that use of a second order state vector would improve matters for straight tracks. However, one would expect that it would still tend to pull tracks into the origin, exhibit less damping than the first order $r\theta$ tracker and have better response to manoeuvres.

For such a system the equation $X'_{k+1} = \Phi X_k$ would expand into:-

$$\begin{bmatrix} r'_{k+1} \\ \dot{r}'_{k+1} \\ \ddot{r}_{k+1} \\ \theta'_{k+1} \\ \dot{\theta}'_{k+1} \\ \ddot{\theta}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & t & \frac{t^2}{2} & o & o & o \\ o & 1 & t & o & o & o \\ o & o & 1 & o & o & o \\ o & o & o & 1 & t & \frac{t^2}{2} \\ o & o & o & o & 1 & \\ o & o & o & o & o & 1 \end{bmatrix} \begin{bmatrix} \hat{r}_k \\ \hat{\dot{r}}_k \\ \ddot{r}_k \\ \hat{\theta}_k \\ \hat{\dot{\theta}}_k \\ \ddot{\theta}_k \end{bmatrix}$$

However, this is still not a perfect representation of a straight track and it has the demerits of complication and the large variances associated with estimates of $\ddot{r}$ and θ_k. We therefore need to transform measurements from r and θ to x and y. This is simple enough:

$x = r \sin \theta$

$y = r \cos \theta$ (θ being interpreted in the nautical sense).

The covariance matrix of r and θ is

$$R_k = \begin{bmatrix} \sigma^2{}_{rr} & o \\ o & \sigma^2{}_{\theta\theta} \end{bmatrix}$$

It is shown in Appendix D that the covariance matrix for x, y is

$$R_{k(x,y)} = \begin{bmatrix} \sigma^2{}_{rr}\mathrm{Sin}^2\theta + r^2\sigma^2{}_{\theta\theta}\mathrm{Cos}^2\theta & (\sigma^2{}_{rr} - r^2\sigma^2{}_{\theta\theta})\mathrm{Sin}\theta\mathrm{Cos}\theta \\ (\sigma^2{}_{rr} - r^2\sigma^2{}_{\theta\theta})\mathrm{Sin}\theta\mathrm{Cos}\theta & \sigma^2{}_{rr}\mathrm{Cos}^2\theta + r^2\sigma^2{}_{\theta\theta}\mathrm{Sin}^2\theta \end{bmatrix}$$

provided that $\sigma_{\theta\theta} \approx \sin\sigma_{\theta\theta}$, which is true for all practical radars.

Hence, all measurements are transformed to Cartesians, the R matrix being transformed as above. The system is then:

$$\begin{bmatrix} x'_{k+1} \\ \dot{x}_{k+1} \\ y'_{k+1} \\ \dot{y}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & t & o & o \\ o & 1 & o & o \\ o & o & 1 & t \\ o & o & o & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \dot{x}_k \\ \hat{y}_k \\ \dot{y}_k \end{bmatrix}$$

11. TESTING RESULTS

a. Each of these three types has been programmed on a computer for comparative testing. A simple simulation has been built round each of the trackers: the simulation generates tracks which consist of two straight parts, the parameters for these parts being provided as input data. Noise is superimposed on each generated plot, the noise being added as range and bearing errors. The noise is produced by a noise generator which produces Gaussian distributed pseudo-random noise (See Appendices E, F and G).

The trackers were tested with various sets of input data. The variances of errors in range and bearing, and the data rate were all kept constant for any one run. This was not due to any limitations of the trackers, but purely in order that the whole system could be set to work quickly. Also, keeping things simple in this manner made it easier to predict what ought to happen.

Some tracks with turns included were generated: it was not expected that performance should be good on turns, since no turn detection or turn handling facilities have been built in. Overshoot on turns was shown by running some of the tracks without noise (the R matrices were not zeroed as the filter would ignore measurements after the first).

b. The results obtained were very much as expected, as may be seen from Figs. 1-15. Both types of polar tracker perform tolerably well with tracks which turn towards the origin, but poor performance against crossing targets. If the distance units are considered as Km and time units seconds, 1 distance unit per time unit corresponds to approx Mach 3. One would expect the $r\theta$ trackers to have poor performance for targets which turn away. Thus the polar trackers tend to assume all tracks are going to approach the origin, whereas the Cartesian trackers do not.

The standard deviations of the plot noise were 0.4 distance units for range measurements, and 0.4° in bearing.

Figure 1 shows the response of the three trackers to a noise free track which consists of two straight parts. All three trackers are satisfactory on the initial radial portion - this is as it should be. The responses to the turn may be clearly seen. If the track had continued beyond the point at which it stopped, the xy tracker would have eventually rejoined it. The purpose of this graph is to show the effect of continually increasing damping in the event of actual motion being significantly different from assumed motion. How to deal with this situation (turn handling) will be dealt with in a later report.

Figures 2-4 show the performances of the trackers against a short noisy track which crosses at 10 distance units. It is clearly seen that the first order polar tracker pulls the track towards the origin - because it is trying to fit the best spiral through all of the points. This effect is less noticeable with the second order polar tracker (Figure 3), and absent with the first order Cartesian Tracker (Figure 4).

Figures 5-15 all have the same speed, 1 distance unit/time unit. Figures 5, 6, 7 and 8 show the relative performances against a straight track with a crossing distance of 30 distance units. The first order $r\theta$ tracker (Figure 6 and 7) is clearly poor in that the incorrect equation of motion introduces systematic errors. The 2nd order $r\theta$ tracker is "reasonable" in removing these systematic errors until the plot at $y = 26.3$, ie to plot number 19 (see Figure 8). However, the xy tracker (Figure 5) is considerably better, both in terms of position and velocity: after plot 6, the xy tracker always has velocity estimates within $\pm$ 0.02 of the true values. When the 2nd order polar tracker applies enough damping to do this, the non-linear assumed equations of motion start to "pull" the track systematically to one side.

Figures 9-15 depict a similar track to that in 5-8, except that the crossing distance has been reduced to 15 distance units. The performance of the xy tracker (Figure 9) is quite reasonable, particularly when Figures 10 and 11 are considered. These two graphs show plotted and smoothed errors together with their σ's and 3σ's. The curves of σ and 3σ for the smoothed errors are approximate, in some later runs the true values will be output during the runs. The first order polar tracker does not perform adequately for very long Figures 12-13). The second order polar tracker performs adequately up to around plot 11.

It may be apparent that limiting the number of plots used in the two $r\theta$ based trackers would have improved performance. In particular, if the track depicted in Figure 11 had a time constant of 11 plots it would probably have maintained good contact. This concept will be pursued further in a later report. It is also worth mentioning that, at the closest point of approach, the equations of motion assumed in polar trackers are valid since motion is circumferential at this point. Of course the above section demonstrates the importance of selecting the correct equations of motion.

12. USE OF THE METHOD USING DIFFERENT MEASURED QUANTITIES

a. We may now make a further point about the usefulness of the Kalman filter in application such as "Bearing only". Consider that we have, in addition to normal radars, a sensor which only provides, say, bearing information. In a polar tracker, we may use this information readily. It may be recalled that M was defined in (19).

For a polar tracker, the state vector, X, may be:

$$\begin{bmatrix} r \\ \dot{r} \\ \theta \\ \dot{\theta} \end{bmatrix}$$; the measured variables may be:

$$Y = \begin{bmatrix} r_y \\ \theta_y \end{bmatrix} \text{;with covariance } R = \begin{bmatrix} \sigma^2_r & o \\ o & \sigma^2_\theta \end{bmatrix} .$$

Hence $M_{r\theta}$ may be seen to be:

$$\begin{bmatrix} 1 & o & o & o \\ o & o & 1 & o \end{bmatrix} .$$

If we only measure bearing, (19) indicates that M_θ would be:

$$[o \quad o \quad 1 \quad o]$$

to give

$$[o \quad o \quad 1 \quad o] \begin{bmatrix} r \\ \dot{r} \\ \theta \\ \dot{\theta} \end{bmatrix} = \theta_y \text{ - noise} .$$

The covariance matrix applicable to θ_y, R_θ is σ^2_θ; a 1x1 matrix.

b. It may now appear difficult to incorporate this single variable measurement in a track represented in Cartesians. However, the fact that we were able to make a bearing measurement at all conveys the information that the range of the target was somewhere between 0 and the maximum range of the sensor. Thus it would not be unreasonable to assign a value of range to the bearing measurement, this range being, say, one half maximum range of some function of signal strength. The R matrix associated with the "measurement" would contain the bearing variance and a variance for range. If the range were better than just a guess, then $\sigma^2{}_r$ could be made reasonable. This would not prejudice the assumption made in Appendix B, since the approximation for this transformation is only effective in bearing.

13. CONCLUSIONS

a. The Kalman filter approach provides the basis for an effective and unified generalised tracking system. Cartesian representation of tracks appears advantageous at this stage. However, when the effects of plant noise are considered, it would appear that this is best applied in a track-oriented manner. Alternatively the knowledge of forecast accuracy could be used in deciding whether or not a turn had occurred - in which case the filter could be reset in some fashion.

b. It has been shown that a system which takes information from various sensors with differing accuracies can be produced and that not all sensors need provide the same sort of information.

c. It is perhaps worth noting that the tracking process can only be as good as the data it receives. Thus it is dependent on the accuracy and rate of input. The amount of use it can make of this information depends on how much of the old information the target being tracked allows us to use. If the targets being followed may only turn slowly, we may use a long time constant to match this, the long time constant giving a high variance reduction ratio ie high gain in precision. "Lively" targets only allow a short time constant which in turn gives a low gain. Consequently, high gain for lively targets requires a higher data rate (ie we make time units smaller). The method should allow us to put figures to such arguments.

d. A long study of the subject of tracking by Bordner and Benedict resulted in them concluding "to improve tracking, improve radars". The approach adopted for this study should provide the basis for quantitative statements on the improvements that new radars could bring about, eg with regard to data rate, accuracy, elevation information, Doppler information etc.

14. FUTURE WORK

No tracking process is of any value without a means of associating extracted plots with tracks. Reference 9 discusses some ideas and methods for such a process. Any such process requires knowledge of accuracies of forecasts and measurements - these will be available if a Kalman filter is used for the tracking process.

Current work on tracking is directed at studying the problems of turn handling, and then, having a solid base, consideration will be given to the benefits and uses of elevation and Doppler information in a generalised manner.

15. REFERENCES

1. "A Method of Processing Radar Plot Data to obtain Position, Velocity and Turn Information" H R Simpson, RRE Memo 1924, July 1962 (UNCLASSIFIED)

2. "Adjustment Rules for Automatic Tracking"
B L Marks, RAE Technical Note Math 79, November 1961 (UNCLASSIFIED).

3. "A proposal for an Automatic T~acking Logic Based on Probability"
S Magowan, RRE Memo 2253, February 1966 (UNCLASSIFIED).

4. P R R Clyn :k and G S Milner May 1969 (Private Commun.cation).

5. "Derivation of the Kalman Filtering Equations from Elementary Statistical Principles"
P M Barham and D E Humphries, RAE Technical Report TR69095, 1969 (UNLIMITED).

6. "A New Approach to Linear Filtering and Prediction Problems"
R E Kalman, Journal of Basic Engineering, Vol 82D, No 1, March 1960 pp 35-45.

7. "New Results in Linear Filtering and Prediction Theory"
R E Kalman and R S Bucy, Journal of Basic Engineering, Vol 83D, No 1, March 1961, pp 95-108.

8. "A Tutorial Derivation of Recursive Weighted Least Squares State Vector Estimation Theory".
J S Pappas, US Army Test and Evaluation Command Report RO-S-68-2, August 1968, AD 674406, (UNCLASSIFIED).

9. "A Theory for Simple Associations During Target Tracking"
A L C Quigley, ASWE Technical Report TR-71-11, 1971 (UNLIMITED)

10. "A Theory for Complex Associations".
A L C Quigley, ASWE Technical Report TR-72-13, 1972 (UNLIMITED)

NOTE: Reports quoted are not necessarily available to members of the public or to commercial organisations.

APPENDIX A

EQUIVALENCE OF KALMAN FILTER, LEAST SQUARES $\alpha\beta$ TRACKERS AND LINEAR REGRESSION, FOR SINGLE DIMENSION CONSTANT ERROR CONSTANT DATA RATE CONDITIONS

Assume measurements are available as follows:

Position	Time
z_1	o
z_2	t
z_3	2t
z_4	3t

All measurements have variance σ^2

1. Kalman estimation: Φ is $\begin{bmatrix} 1 & t \\ o & 1 \end{bmatrix}$, M is $[1 \quad o]$

$$\begin{bmatrix} x'_n \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 1 & t \\ o & 1 \end{bmatrix} \begin{bmatrix} x_{n-1} \\ \dot{x}_{n-1} \end{bmatrix}$$

Initially, let $x_1 = z_1$, $\dot{x}_1 = o$, velocity variance large

Hence X_2 is $\begin{bmatrix} z_2 \\ \dfrac{z_2 - z_1}{t} \end{bmatrix}$

$$P_2 = \begin{bmatrix} \sigma^2 & \dfrac{\sigma^2}{t} \\ \dfrac{\sigma^2}{t} & \dfrac{2\sigma^2}{t^2} \end{bmatrix}$$

$$x'_3 = \begin{bmatrix} 2z_2 - z_1 \\ \dfrac{z_2 - z_1}{t} \end{bmatrix}$$

$$P'_3 = \begin{bmatrix} 5\sigma^2 & \dfrac{3\sigma^2}{t} \\ \dfrac{3\sigma^2}{t} & \dfrac{2\sigma^2}{t^2} \end{bmatrix}$$

$$K_3 = \begin{bmatrix} \dfrac{5}{6} \\ \dfrac{1}{2t} \end{bmatrix}$$

$$\hat{X}_3 = \begin{bmatrix} \frac{5}{6}z_3 + \frac{z_2}{2} - \frac{z_1}{6} \\ \frac{(z_3-z_1)}{2t} \end{bmatrix}$$

$$P_3 = \begin{bmatrix} \frac{5\sigma^2}{6} & \frac{\sigma^2}{2t} \\ \frac{\sigma^2}{2t} & \frac{\sigma^2}{2t^2} \end{bmatrix}$$

$$X'_4 = \begin{bmatrix} \frac{1}{3}(4z_3+z_2-2z_1) \\ \frac{(z_3-z_1)}{2t} \end{bmatrix}$$

$$P'_4 = \begin{bmatrix} \frac{7\sigma^2}{3} & \frac{\sigma^2}{t} \\ \frac{\sigma^2}{t} & \frac{\sigma^2}{2t^2} \end{bmatrix} \qquad K_4 = \begin{bmatrix} \frac{7}{10} \\ \frac{3}{10t} \end{bmatrix}$$

$$\hat{X}_4 = \begin{bmatrix} \frac{1}{10}(7z_4+4z_3+z_2-2z_1) \\ \frac{1}{10t}(3z_4+z_3-z_2-3z_1) \end{bmatrix}$$

2. Linear Regression Estimation

We assume observations are related thus:

$$x = mt+c$$

We thus wish to minimise $s = \frac{1}{n}\sum_{i=1}^{n}(z_i-mt_i-c)^2$

We do this by finding $\frac{\partial s}{\partial m}, \frac{\partial s}{\partial c}$, setting both to zero and solving for m and c.

We get $$m = \frac{n\Sigma zt - \Sigma z\Sigma t}{n\Sigma t^2 - (\Sigma t)^2}$$

$$c = \frac{\Sigma t^2\Sigma z - \Sigma tz\Sigma t}{n\Sigma t^2 - (\Sigma t)^2}$$

Hence, in this example, $n = 4$ and z's and times are as given, (ie o, t, 2t etc)

$$\therefore \quad m = \frac{1}{10t}(3z_4+z_3-z_2-3z_1)$$

$$c = \frac{1}{10}(7z_1+4z_2+z_3-2z_4)$$

$\therefore$ Position at time 3t is:

$$\hat{x}_4 = \frac{1}{10t}(3z_4+z_3-z_2-3z_1).\ 3t + \frac{1}{10}(7z_1+4z_2+z_3-2z_4)$$

$$= \frac{1}{10}(7z_4+4z_3+z_2-2z_1)$$

and its velocity is m.

3. Least squares αβ tracker

Now, $\alpha = \dfrac{2(2n-1)}{n(n+1)}$ and $\beta = \dfrac{6}{n(n+1)}$

Thus $\dot{x}_2 = \dfrac{z_2-z_1}{t}$

and $\hat{x}_2 = z_2$

$\therefore$ $x'_3 = z_2+(z_2-z_1)$

$\therefore$ $\hat{x}_3 = x'_3 + \frac{5}{6}(z_3-x'_3) = \frac{1}{6}(-z_1+2z_2+5z_3)$

$\therefore$ $\dot{x}_3 = \dot{x}_2 + \frac{1}{2t}(z_3-x'_3) = \frac{1}{2t}(-z_1+z_3)$

$\therefore$ $x'_4 = \hat{x}_3 + \dot{x}_3.t = \frac{1}{3}(-2z_1+z_2+4z_3)$

$\therefore$ $\hat{x}_4 = x'_4 + \frac{7}{10}(z_4-x'_4) = \frac{1}{10}(-2z_1+z_2+4z_3+7z_4)$

and $\dot{x}_4 = \dot{x}_3 + \frac{3}{10t}(z_4-x'_4) = \frac{1}{10t}(-3z_1-z_2+z_3+3z_4)$

Hence all three methods give the same results, which is not surprising since they all originate from the same criterion. Proofs which demonstrate a difference between linear regression and least squares αβ tracking must therefore contain an error of some sort.

APPENDIX B

TRANSFORMATION OF VECTORS AND THEIR COVARIANCES

The problem discussed here is the one which arises when we transform a given vector in one plane to another plane. We wish to find how the covariance matrix which applies to the original vector is transformed.

Consider that we have a set of n variables, which we group into a vector Y, together with a covariance matrix R_Y which is nxn and contains the covariance of the variables in Y. We will transform Y to a new vector X and we now wish to find the contents of a matrix R_x, which is nxn and contains the covariances applicable to X.

$$Y = \begin{bmatrix} y_1 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix}, \qquad X = \begin{bmatrix} x_1 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix}$$

$$R_Y = \begin{bmatrix} r_{y11} & r_{y12} & \cdot & \cdot & \cdot & r_{y1n} \\ \cdot & \cdot & & & & \\ \cdot & & & & & \\ \cdot & \cdot & & & & \\ \cdot & \cdot & & & & \\ \cdot & & & & & \\ \cdot & \cdot & & & & \\ r_{yn1} & r_{yn2} & \cdot & \cdot & \cdot & r_{ynn} \end{bmatrix}$$

$$R_x = \begin{bmatrix} r_{x11} & r_{x12} & \cdot & \cdot & \cdot & r_{x1n} \\ \cdot & & & & & \\ \cdot & & & & & \\ \cdot & & & & & \\ \cdot & & & & & \\ r_{xn1} & r_{xn2} & \cdot & \cdot & \cdot & r_{xnn} \end{bmatrix}$$

By definition,

$$r_{xik} = \lim_{m\to\infty} \frac{1}{m} \sum_{q=1}^{m} (x_{iq} - \overline{x}_i)(x_{kq} - \overline{x}_k)$$

or, in increments from the mean values $\overline{x}_i$ and $\overline{x}_k$

$$r_{xik} = \lim_{m\to\infty} \frac{1}{m} \sum_{q=1}^{m} \delta x_{iq}\, \delta x_{kq} \qquad \text{(B.1)}$$

From the total derivative theorem,

$$dx_i = \frac{\partial x_i}{\partial y_1} dy_1 + \frac{\partial x_i}{\partial y_2} dy_2 + \ldots\ldots + \frac{\partial x_i}{\partial y_n} dy_n$$

$$= \sum_{j=1}^{n} \frac{\partial x_i}{\partial y_j} dy_j \qquad (B.2)$$

Now, if the derivatives of Y are linear over the intervals $\delta y_s = \sqrt{r_{sss}}$ $s = 1,2,3 \ldots n$, we may use (B.1) and (B.2) to give:

$$r_{xik} = \lim_{m\to\infty} \frac{1}{m} \sum_{q=1}^{m} \left[\left(\sum_{j=1}^{n} \frac{\partial x_i}{\partial y_j} \delta y_{jq} \right) \left(\sum_{s=1}^{n} \frac{\partial x_k}{\partial y_k} \delta y_{sq} \right) \right]$$

$$= \lim_{m\to\infty} \frac{1}{m} \sum_{q=1}^{m} \sum_{j=1}^{n} \sum_{s=1}^{n} \frac{\partial x_i}{\partial y_j} \frac{\partial x_k}{\partial y_s} \delta y_{jq} \delta y_{sq}$$

$$= \sum_{j=1}^{n} \sum_{s=1}^{n} \frac{\partial x_i}{\partial y_j} \frac{\partial x_k}{\partial y_s} \left(\lim_{m\to\infty} \frac{1}{m} \sum_{q=1}^{m} \delta y_{jq} \delta y_{sq} \right)$$

These last two steps are valid for these summations.

The last part (ie following $\lim_{m\to\infty}$) in the expression for r_{xik} is the definition of the covariance of y_j and y_s hence:

$$r_{xik} = \sum_{j=1}^{n} \sum_{s=1}^{n} \left(\frac{\partial x_i}{\partial y_j} \frac{\partial x_k}{\partial y_s} r_{yjs} \right) \qquad (B.3)$$

We have therefore established the relationship between R_y, the covariance matrix of Y, and R_x, the covariance matrix of X.

This may be written in matrix form as:-

$$R_x = AR_yA^T$$

where $A_{ij} = \frac{\partial x_i}{\partial y_j}$

APPENDIX C

ESTIMATION OF MOST PROBABLE POSITION

The joint probability density function is given by (see text 9 a.)

$$p = \frac{1}{\sqrt{2\pi r\ a'}} \cdot \exp \left\{ - \frac{(x'_2-x)^2}{2a'} - \frac{(x_2-x)^2}{2r} \right\}$$

$$= \frac{1}{\sqrt{2\pi r\ a'}} \cdot \exp (J)$$

The value of x, $\hat{x}$ which maximises p is given by the solution of $\frac{dJ}{dx} = 0$

Hence $$\frac{dJ}{dx} = \frac{x'_2-x}{a'} + \frac{x_2-x}{r} = 0$$

$$\therefore \quad \hat{x} = \frac{rx'_2+a'x_2}{a'+r}$$

We also wish to know the variance of $\hat{x}$ which we call σ^2

Let $h = \frac{r}{a'+r}$, $k = \frac{a'}{a'+r}$

Then $$\hat{x} = hx'_2 + kx_2$$

Let us assume an error of $\delta x'_2$ in x'_2 and δx_2 in x_2.

Then the error in $\hat{x}$, $\delta\hat{x} = h.\delta x'_2 + k.\delta x_2$

The variance of $\hat{x}$ is defined as the mean value of $(\delta\hat{x})^2$ over many samples.

Thus $\sigma^2 = \text{mean } (h^2(\delta x'_2)^2 + 2hk\delta x'_2\delta x_2 + k^2(\delta x_2)^2)$.

We assume that measurement errors are uncorrelated, ie that $\delta x'_2$ and δx_2 are independent; we also assume that they have zero mean. Hence mean $(\delta x'_2\delta x_2) = 0$

Thus $$\sigma^2 = h^2a' + k^2r$$

$$= \frac{r^2a' + r(a')^2}{(a'+r)^2} = \frac{ra'(r+a')}{(r+a')^2}$$

$$= \frac{ra'}{r+a'}$$

ESTIMATION OF MOST PROBABLE VELOCITY

The method of estimating velocity is basically the same as that for estimating position with one important difference. Our two estimates of velocity are:

$$v_1 = \frac{x'_2 - \hat{x}_1}{t}$$

and $$v_2 = \frac{x_2 - \hat{x}_1}{t} .$$

It is clear that the two estimates are related since they both include data from the same point. If we call these estimates v_1 and v_2 respectively, and assume variances s_1^2 and s_2^2 and covariances s_{12}^2 (these values are calculated in C.2) then the joint probability density function may be shown to be given by:

$$p = \frac{1}{2\pi s_1 s_2 \sqrt{1-\rho^2}} \cdot \exp \left\{ \frac{-\frac{(v_1-v)^2}{s_1^2} - \frac{(v_2-v)^2}{s_2^2} + \frac{2\rho(v-v_1)(v-v_2)}{s_1 s_2}}{2(1-\rho^2)} \right\}$$

where $$\rho = \frac{s^2{}_{12}}{s_1 s_2}$$

We wish to find the value of v which maximises p ie the value of v to satisfy

$\frac{dJ}{dv} = 0$, where J is the argument of the above exponential function.

$$\frac{dJ}{dv} = \frac{\frac{2(v_1-v)}{s_1^2} + \frac{2(v_2-v)}{s_2^2} + \frac{2\rho(2v-v_1-v_2)}{s_1 s_2}}{2(1-\rho^2)} = 0$$

Hence $$v = \frac{s_2^2 v_1 + s_1^2 v_2 - s_{12}^2(v_1+v_2)}{s_1^2+s_2^2-2s_{12}^2} \qquad \text{(C.1)}$$

Now we know that $v_2 = \frac{x_2-\hat{x}_1}{t}$ and $v_1 = \frac{x'_2-\hat{x}_1}{t}$

which may be expressed in matrix form as:

V = FX ie

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} -\frac{1}{t} & \frac{1}{t} & 0 \\ -\frac{1}{t} & 0 & \frac{1}{t} \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ x'_2 \\ x_2 \end{bmatrix}$$

If we consider errors in each element of X, we may construct a matrix out of them:

$$\begin{bmatrix} \delta\hat{x}_1 \\ \delta x'_2 \\ \delta x_2 \end{bmatrix}$$

Now the variance in $\hat{x}_1$ is the mean value over many samples of $(\delta\hat{x}_1)^2$, and so on for x'_2 and x_2. The covariance of $\hat{x}_1$ and x'_2 is the mean over many samples, of $(\delta\hat{x}_1.\delta x'_2)$. We may represent these variances and covariances in a covariance matrix, which contains the mean values of the elements in:

$$\begin{bmatrix}\delta\hat{x}_1\\ \delta x'_2\\ \delta x_2\end{bmatrix}\begin{bmatrix}\delta\hat{x}_1 & \delta x'_2 & \delta x_2\end{bmatrix}$$

ie the covariance matrix elements are the mean values of the elements in:

$$\begin{bmatrix}(\delta\hat{x}_1)^2 & \delta\hat{x}_1\delta x'_2 & \delta\hat{x}_1\delta x_2\\ \delta x'_2\delta\hat{x}_1 & (\delta x'_2)^2 & \delta x'_2\delta x_2\\ \delta x_2\delta\hat{x}_1 & \delta x_2\delta x'_2 & (\delta x_2)^2\end{bmatrix}$$

Now the covariance matrix associated with X is

$$\begin{bmatrix}a & p & o\\ p & a' & o\\ o & o & r\end{bmatrix}$$

where a is the variance of $\hat{x}_1$, r is the variance of x_2, p is the covariance of $\hat{x}_1$ and x'_2, and a' is the variance of x'_2. Note that the covariances associated with x_2 are zero, since we assume that errors in this measurement are not related in any way to errors in previous measurements.

We may apply (B.3) of Appendix B to calculate the covariance matrix of V, which is:

$$\begin{bmatrix}s_1{}^2 & s_{12}{}^2\\ s_{12}{}^2 & s_2{}^2\end{bmatrix} = \begin{bmatrix}\frac{a+a'-2p}{t^2} & \frac{a-p}{t^2}\\ \frac{a-p}{t^2} & \frac{a+r}{t^2}\end{bmatrix} \tag{C.2}$$

We may apply a similar approach to estimate p ie

given $\quad x'_2 = \begin{bmatrix}1 & t\end{bmatrix}\begin{bmatrix}\hat{x}_1\\ v_1\end{bmatrix}$

with covariance for $\begin{bmatrix}\hat{x}_1\\ v_1\end{bmatrix}$ of $\begin{bmatrix}a & b\\ b & d\end{bmatrix}$

and applying (B.3) we get, for covariance of

$$\begin{bmatrix}\hat{x}_1\\ x'_2\end{bmatrix}:$$

$$\begin{bmatrix} a & p \\ p & a' \end{bmatrix} = \begin{bmatrix} a & a+bt \\ a+bt & a+2bt+t^2d \end{bmatrix}$$

Thus $\quad P = a+bt$

We may thus combine this result with (C.2) and (C.1) to give:

$$v = \frac{(x'_2-\hat{x}_1)(a+r+bt) + (x_2-\hat{x}_1)(t^2d+tb)}{t(a+2bt+t^2d+r)}$$

Now

$$\frac{a+r+bt}{t(a+2bt+t^2d+r)} = \frac{1}{t} - \frac{(t^2d+bt)}{t(a+2bt+t^2d+r)}$$

We may thus rearrange (C.2) to give

$$v = \frac{x'_2-\hat{x}_1}{t} + \frac{(bt+t^2d)(x_2-x'_2)}{t(a+2bt+t^2d+r)}$$

$$= v_1 + \frac{(b+td)(x_2-x'_2)}{a+2bt+t^2d+r} \qquad \text{(C.3)}$$

v_1 being the previous estimate of velocity.

We now move on to calculate the variance of v which we will call s^2.

Now, $\quad v = v_1 + \frac{\beta}{t}(x_2-x_2{}^1).$

Thus we are performing the transformation:

$$v = [\tfrac{\beta}{t} \;\; -\tfrac{\beta}{t} \;\; 1] \begin{bmatrix} x_2 \\ x'_2 \\ v_1 \end{bmatrix}.$$

In order to find the variance of v, we need to determine the covariance matrix associated with

$$\begin{bmatrix} x_2 \\ x_2' \\ v_1 \end{bmatrix}.$$

We assume that x_2 has variance r, and is uncorrelated with x_2' and v_1 (ie that errors in measurement x_2 are not related to errors in forecast and velocity estimates). Thus all that remains to be estimated is the covariance matrix for

$$\begin{bmatrix} x_2' \\ v_1 \end{bmatrix}$$

We may derive this by applying (B.3) to the transformation:

$$x_2' = \hat{x}_1 + v_1 . t$$

$$v_1 = v_1$$

giving the covariance matrix $\begin{bmatrix} a+2bt+t^2d & b+td \\ b+td & d \end{bmatrix}$

Hence the covariance matrix for $\begin{bmatrix} x_2 \\ x_2' \\ v_1 \end{bmatrix}$ is $\begin{bmatrix} r & o & o \\ o & a+2bt+t^2d & b+td \\ 0 & b+td & d \end{bmatrix}$

Thus we may use (B.3) to calculate s^2:

$$s^2 = \frac{\beta^2 r}{t^2} + \frac{\beta^2}{t^2} (a+2bt+t^2d) + d - \frac{2\beta}{t} (b+td)$$

Now $\frac{\beta}{t} = \frac{(b+td)}{a+2bt+t^2d+r}$ (as shown earlier)

$$\therefore \quad s^2 = \frac{(b+td)^2 [r+a+2bt+t^2d-2(a+2bt+t^2d+r)]}{(a+2bt+t^2d+r)^2} + d$$

$$= \frac{d(a+2bt+t^2d+r)^2 - (b+td)^2(a+2bt+t^2d+r)}{(a+2bt+t^2d+r)^2}$$

$$\therefore \quad s^2 = \frac{ad+dr-b^2}{a+2bt+t^2d+r}$$

$$\therefore \quad s^2 = \frac{d(a+r)-b^2}{a+2bt+t^2d+r}$$

Summary

The best estimate of position has been shown to be given by:

$$\hat{x} = \frac{rx_2' + a'x_2}{a'+r}$$

and its variance is:

$$\sigma^2 = \frac{ra'}{r+a'}$$

where $\quad a' = a+2bt+t^2d$

The best estimate of velocity is given by:

$$v = v_1 + \frac{(b+td)(x_2 - x_2')}{a+2bt+t^2d+r}$$

and its variance is

$$s^2 = \frac{d(a+r)-b^2}{a+2bt+t^2d+r}$$

APPENDIX D

APPLICATION OF RESULTS IN APPENDIX B IN POLAR TO CARTESIAN TRANSFORMATION

Consider the following polar to Cartesian transformation (ie from r, θ to x, y)

$$x = r \sin \theta$$

$$y = r \cos \theta$$

(Note that θ is interpreted in the nautical sense ie θ is a bearing which increases in a clockwise direction from the positive y axis.)

The covariance matrix associated with r and θ is assumed to be:

$$\begin{bmatrix} \sigma^2{}_{rr} & 0 \\ 0 & \sigma^2{}_{\theta\theta} \end{bmatrix} \quad \text{ie r and } \theta \text{ are independently variable}$$

(as would be the case when r and θ are determined from a radar auto-detection system).

We may relate the problem to the argument in Appendix B by considering:

$$X \text{ as } \begin{bmatrix} x \\ y \end{bmatrix}$$

$$Y \text{ as } \begin{bmatrix} r \\ \theta \end{bmatrix}$$

The transformation of covariances given by (B.3) involved a condition of linearity, this holds if $\sin \sigma_{\theta\theta} \simeq \sigma_{\theta\theta}$ which is valid for all radars the author is aware of.

We will denote the covariance matrix applicable to X as:

$$Rx = \begin{bmatrix} \sigma^2{}_{xx} & \sigma^2{}_{xy} \\ \sigma^2{}_{xy} & \sigma^2{}_{yy} \end{bmatrix}$$

Hence, by substitution into (B.3)

$$\sigma^2_{xx} = \left(\frac{\partial x}{\partial r}\right)^2 \sigma^2_{rr} + \left(\frac{\partial x}{\partial \theta}\right)^2 \sigma^2_{\theta\theta} \qquad (\text{since } \sigma^2_{r\theta} = \sigma^2_{\theta r} = 0)$$

$$= \sigma^2_{rr} \sin^2\theta + r^2\, \sigma^2_{\theta\theta} \cos^2\theta$$

$$\sigma^2_{xy} = \sigma^2_{yx} = \frac{\partial x}{\partial r}\frac{\partial y}{\partial r}\sigma^2_{rr} + \frac{\partial x}{\partial \theta}\frac{\partial y}{\partial \theta}\sigma^2_{\theta\theta}$$

$$= (\sigma^2_{rr} - r^2\sigma^2_{\theta\theta}) \sin\theta \cos\theta$$

$$\sigma^2_{yy} = \left(\frac{\partial y}{\partial r}\right)^2 \sigma^2_{rr} + \left(\frac{\partial y}{\partial \theta}\right)^2 \sigma^2_{\theta\theta}$$

$$= \sigma^2_{rr} \cos^2\theta + r^2\sigma^2_{\theta\theta} \sin^2\theta$$

$$\text{Hence } Rx = \begin{bmatrix} \sigma^2_{rr} \sin^2\theta + r^2\sigma^2_{\theta\theta}\cos^2\theta & (\sigma^2_{rr} - r^2\sigma^2_{\theta\theta})\sin\theta\cos\theta \\ (\sigma^2_{rr} - r^2\sigma^2_{\theta\theta})\sin\theta\cos\theta & \sigma^2_{rr}\cos^2\theta + r^2\sigma^2_{\theta\theta}\sin^2\theta \end{bmatrix}$$

This is the covariance matrix for the transformed vector X which we set out to find.

It is interesting to note that in general, the co-ordinates after the transformation (x and y) are not independent. The singular cases where they are independent occur for $\theta = 0^o, 90^o, 180^o, \ldots$ or for $r = \sigma_{rr}/\sigma_{\theta\theta}$. These special cases are easily explained as follows: the equal probability contours for bivariate Gaussian distributions are ellipses: if the axes lie in the directions of the axes of the frame of reference, then the covariance terms are zero: hence the independence of x and y errors for $\theta = 0^o, 90^o$ etc. When $r = \sigma_{rr}/\sigma_{\theta\theta}$, these ellipses reduce to circles which may be considered as limiting ellipses whose major and minor axes lie along the axes of the reference frame.

It is worth pointing out that the covariance matrix R_x could have been derived by considering the geometry of the ellipse and the rotation required after transformation. However, this is complicated, particularly for a transformation from one 3-D frame to another, when it is not just a simple translation or rotation of axes.

APPENDIX E

XY FIRST ORDER TRACKER - PROGRAM HARD COPY

```
*IODUNIT10/PRINT
*IODUNIT20/READ
*XEQ
*STORAGE/4000/15000
*CHAIN1
*ALGOL
   'BEGIN 'INTEGER' I, J,MEASURE, STATE,MI,N,O,OLDX¢
     'ARRAY 'X(1:4,1:1),R(1:2,1:2),Z(1:2,1:1),
     P(1:4,1:4),M(1:2,1:4),MT(1:4,1:2),
     PHI(1:4,1:4), K(1:4,1:2),PMT(1:4,1:2),OUTX(1:4,1:1),
     OUTZ(1:2,1:1) ¢
     'REAL ' T, TIME, DELTAT, CHANGE, FINISH, FX, XDOT, CHANGE OF XDOT,
     WY, YDOT, CHANGE OF YDOT, X0, Y0, VARR, VARTHETA, RNOISE, THETANOISE,
     U1, U2, VARIANCE, NOISE, RANGE, BEARING, SINTHETA, COSTHETA¢

     'COMMENT' X,Z AND COPYZ ARE COLUMN VECTORS.
     R AND COPYR ARE 2*2 MEASUREMENT COVARIANCE MATRICES.
     K IS A 4*2 MATRIX, AND IS ESSENTIALLY THE DAMPING FACTOR.
     M IS THE MEASUREMENT MATRIX AND IS 2*4.
     MT IS THE TRANSPOSE OF M.
     TRACK IN X,Y,GENERATED IN X,Y WITH CONSTANT DATA RATE.
     MANOEUVRE AT TIME CHANGE. FINISH AT TIME FINISH.
     KALMAN FILTER TRACKER WITHOUT PLANT NOISE TERMS ¢

      'PROCEDURE' TRANSPOSE(PHI,PHIT,MI,N) ¢
      'INTEGER' MI,N ¢
      'ARRAY' PHI,PHIT ¢
      'BEGIN 'INTEGER' I,J ¢
         'ARRAY' COPYPHI(1:MI,1:N) ¢
         'FOR'I=1 'STEP'1 'UNTIL'MI 'DO'
         'FOR'J=1 'STEP'1 'UNTIL'N 'DO'
         COPYPHI(I,J)=PHI(I,J) ¢
         'COMMENT' PHI COPIED SO THAT TRANSPOSE CAN BE ASSIGNED
         TO ORIGINAL MATRIX(ONLY POSSIBLE FOR SQUARE MATRICES) ¢
         'FOR'I=1 'STEP'1 'UNTIL'MI 'DO'
         'FOR'J=1 'STEP'1 'UNTIL'N 'DO'
         PHIT(J,I)=COPYPHI(I,J) ¢
      'END'TRANSPOSE ¢

      'PROCEDURE' MATRIXINV(A,B,MI) ¢
      'ARRAY' A,B ¢
      'INTEGER' MI ¢
      'BEGIN 'REAL' DET ¢
      'COMMENT' TO INVERT A 2*2 MATRIX ¢
         'IF'MI 'NE'2'THEN 'BEGIN'
            WRITETC(10, '(('C') 'M NE 2') ') ¢
            'GOTO' STOP ¢
         'END' ¢
         DET=A(1,1)*A(2,2)-A(1,2)*A(2,1) ¢
         B(1,1)=A(2,2)/DET ¢
         B(1,2)=(-A(1,2)/DET) ¢
         B(2,1)=(-A(2,1)/DET) ¢
         B(2,2)=A(1,1)/DET ¢
      STOP: 'END' MATRIXINV ¢
```

```
'PROCEDURE' MATRIXADD(A,B,C,MI,N) ¢
'ARRAY' A,B,C ¢
'INTEGER' MI,N ¢
'COMMENT' THE SUM OF THE MATRICES IS STORED IN THE ARRAY
C.A,B,C ARE MI*N ¢
'BEGIN' 'INTEGER' I,J,P ¢
    'FOR'I=1 'STEP'1 'UNTIL'MI 'DO'
    'FOR'J=1 'STEP'1 'UNTIL'N 'DO'
   C(I,J)=A(I,J)+B(I,J) ¢
'END' MATRIXADD ¢

'PROCEDURE' MATRIXSUB(A,B,C,MI,N) ¢
'ARRAY' A,B,C ¢
'INTEGER' MI,N ¢
'COMMENT' THE DIFFERENCE OF THE TWO MATRICES IS STORED IN THE
ARRAY C. A,B,C ARE MI*N ¢
'BEGIN' 'INTEGER' I,J ¢
    'FOR'I=1 'STEP'1 'UNTIL'MI 'DO'
    'FOR'J=1 'STEP'1 'UNTIL'N 'DO'
   C(I,J)=A(I,J)-B(I,J) ¢
'END' MATRIXSUB ¢

'PROCEDURE' MATRIXMULT(A,B,C,MI,N,P) ¢
'ARRAY' A,B,C ¢
'INTEGER' MI,N,P ¢
'COMMENT' THE PRODUCT OF THE TWO ARRAYS IS STORED IN THE
ARRAY C. TO MULTIPLY BY A SCALAR THE SECOND
MATRIX MU., HAVE THE VALUE OF THE SCALAR AS ITS DIAGONAL
WITH THE REST OF THE ELEMENTS AS NOUGHTS. A IS M*N,B IS N*P,
C IS M*P. COPIES OF INPUT ARRAYS ARE MADE IN ORDER
THAT AN ARRAY MAY BE USED BOTH AS AN INPUT AND
OUTPUT PARAMETER ¢
'BEGIN' 'INTEGER' I,J,X ¢
    'ARRAY' COPYA(1:MI,1:N),COPYB(1:N,1:P) ¢
    'FOR'I=1 'STEP'1 'UNTIL'MI 'DO'
    'FOR'J=1 'STEP'1 'UNTIL'N 'DO'
   COPYA(I,J)=A(I,J) ¢
    'FOR'I=1 'STEP'1 'UNTIL'N 'DO'
    'FOR'J=1 'STEP'1 'UNTIL'P 'DO'
   COPYB(I,J)=B(I,J) ¢
    'FOR'I=1 'STEP'1 'UNTIL'MI 'DO'
    'FOR'J=1 'STEP'1 'UNTIL'P 'DO'
    'BEGIN' C(I,J)=0 ¢
        'FOR'X=1 'STEP'1 'UNTIL'N 'DO'
       C(I,J)=COPYA(I,X)*COPYB(X,J)+C(I,J) ¢
    'END' ¢
'END' MATRIXMULT ¢
```

```
'PROCEDURE' DUMP(A,MI,N) $
'ARRAY' A $
'INTEGER' MI,N $
'BEGIN INTEGER'I,J $
   'FOR'I=1'STEP'1'UNTIL'MI'DO'
   'BEGIN'
      'FOR'J=1'STEP'1'UNTIL'N'DO'
      WRITE(10,LAYOUT('('S-NDDDDD.DDD')'),A(I,J)) $
   'END' $
   NEWLIN(10,4) $
'END' DUMP $


'REAL PROCEDURE' RANDOM $
'BEGIN INTEGER' CALC $
'COMMENT' OLDX MUST BE DECLARED GLOBALLY $
   OLDX=255*OLDX $
   CALC=ENTIER(OLDX/131071) $
   OLDX=OLDX-131071*CALC $
   RANDOM=OLDX/131071 $
'END' RANDOM $

'PROCEDURE' GAUSSIAN(NOISE,VARIANCE) $
'REAL 'NOISE,VARIANCE$
'BEGIN REAL' TWOPI $
   TWOPI=6 28318 $
   U1=RANDOM $
   U2=RANDOM $
   NOISE=SQRT(-2*VARIANCE* LN(U1))*COS(TWOPI*U2) $
'END' GAUSSIAN $

'PROCEDURE'SET R MATRIX $
'BEGIN'
  R(1,1)=SINTHETA*SINTHETA*VARR+Z(2,1)*Z(2,1)*VARTHETA $
  R(1,2)=R(2,1)=SINTHETA*COSTHETA*(VARR-
     RANGE*RANGE*VARTHETA) $
  R(2,2)=COSTHETA*COSTHETA*VARR
     +Z(1,1)*Z(1,1)*VARTHETA $
'END' SET R MATRIX $

'PROCEDURE' NOISEPLOT $
'BEGIN COMMENT' TO SET Z MATRIX WHEN NOISE ADDED $
   THETANOISE=RNOISE=0 $
   GAUSSIAN(RNOISE,VARR) $
   GAUSSIAN(THETANOISE,VARTHETA) $
  RANGE=SQRT(EX*EX+WY*WY)+RNOISE $
  BEARING=ARCTAN(EX/WY)+THETANOISE $
  SINTHETA=SIN(BEARING) $ COSTHETA=COS(BEARING) $
  Z(1,1)=RANGE*SINTHETA $ Z(2,1)=RANGE*COSTHETA $
  SET R MATRIX $
'END' NOISEPLOT $
```

```
'PROCEDURE' CLEANPLOT $
'BEGIN'
  RANGE=SQRT(EX*EX+WY*WY) $
  BEARING=ARCTAN(EX/WY) $
  SINTHETA=SIN(BEARING) $ COSTHETA=COS(BEARING) $
  Z(1,1)=EX $ Z(2,1)=WY $
  SET R MATRIX $
'END' CLEANPLOT $

'PROCEDURE' TRACK(MEASURE,STATE,PHI,M,MT,P,X,Z,R) $
'ARRAY' PHI,P,X,Z,R,MT,M $
'INTEGER' MEASURE,STATE $
'BEGIN' ARRAY' MP(1:MEASURE,1:STATE),PHIT(1:STATE,1:STATE),
MPMT(1:MEASURE,1:MEASURE),PREDZ(1:MEASURE,1:1),
ERRORCOV(1:MEASURE,1:MEASURE),CORRECTION(1:STATE,1:1),
KMP(1:STATE,1:STATE) $
   TRANSPOSE(PHI,PHIT,4,4) $
   MATRIXMULT(PHI,X,X,4,4,1) $
   MATRIXMULT(P,PHIT,P,4,4,4) $
   MATRIXMULT(PHI,P,P,4,4,4) $
   MATRIXMULT(M,P,MP,2,4,4) $
   MATRIXMULT(P,MT,PMT,4,4,2) $
   MATRIXMULT(M,PMT,MPMT,2,4,2) $
   MATRIXADD(MPMT,R,MPMT,2,2) $
   MATRIXINV(MPMT,ERRORCOV,2) $
   MATRIXMULT(PMT,ERRORCOV,K,4,2,2) $
   MATRIXMULT(M,X,PREDZ,2,4,1) $
   MATRIXSUB(PREDZ,Z,PREDZ,2,1) $
   MATRIXMULT(K,PREDZ,CORRECTION,4,2,1) $

   'COMMENT' PREDZ IS NOW (M*X-Z)*K $

   MATRIXSUB(X,CORRECTION,X,4,1) $
   MATRIXMULT(K,MP,KMP,4,2,4) $
   MATRIXSUB(P,KMP,P,4,4) $

   'COMMENT' FOR A 4 VARIABLE STATE VECTOR(STATE=4),
   AND A TWO VARIABLE MEASUREMENT VECTOR(MEASURE=2),
   X IS 4*1,Z IS 2*1, P IS 4*4, M IS 2*4, PHI IS 4*4,K IS 4*2 $
'END' TRACK $

M(1,1)=M(2,3)=MT(1,1)=MT(3,2)=1 $
M(1,2)=M(1,3)=M(1,4)=M(2,1)=M(2,2)=M(2,4)=MT(1,2)=MT(2,2)=MT(2,1)=
MT(3,1)=MT(4,1)=MT(4,2)=0 $
'FOR'I=1'STEP'1'UNTIL'4'DO'
'FOR'J=1'STEP'1'UNTIL'4'DO'
'BEGIN'
   PHI(I,J)='IF'I'EQ'J'THEN'1'ELSE'0 $
   P(I,J)=0 $
'END' $
STATE=4 $  MEASURE=2 $

'COMMENT' STATE VECTOR IS 4-VARIABLE,MEASUREMENT VECTOR
IS 2-VARIABLE $
```

```
  X0=READ(20) $  Y0=READ(20) $
  XDOT=READ(20) $  YDOT=READ(20) $
  CHANGE OF XDOT=READ(20) $  CHANGE OF YDOT=READ(20) $
  CHANGE=READ(20) $  FINISH=READ(20) $  DELTAT=READ(20) $
  VARR=READ(20) $  VARTHETA=READ(20) $  OLDX=READ(20) $
  NOISE=READ(20) $

  WRITET(10,'(('C')'X'*'Y'*'KALMAN'*'TRACKER'('C'))') $
INITIATE TRACK:EX=X0 $ WY=Y0 $
   'IF'NOISE 'EQ' 1 'THEN' NOISEPLOT $
   'IF'NOISE 'EQ' 0 'THEN' CLEANPLOT $
    X(1,1)=Z(1,1) $ X(3,1)=Z(2,1) $
    X(2,1)=X(4,1)=0 $
    DUMP(X,4,1) $ DUMP(Z,2,1) $
    P(1,1)=R(1,1) $ P(3,3)=R(2,2) $
    P(1,3)=P(3,1)=R(1,2) $
    P(2,2)=P(4,4)=1 $
    'COMMENT' TRACK IS NOW PRIMED $
    'FOR' T=DELTAT 'STEP' DELTAT 'UNTIL' CHANGE 'DO'
   'BEGIN'
    EX=X0+XDOT*T $ WY=Y0+YDOT*T $
    'IF' NOISE 'EQ' 1 'THEN' NOISEPLOT $
    'IF' NOISE 'EQ' 0 'THEN' CLEANPLOT $
    'COMMENT' NOISE ADDED IN NOISEPLOT $
    PHI(1,2)=PHI(3,4)=DELTAT $
    TRACK(2,4,PHI,M,MT,P,X,Z,R) $
    DUMP(X,4,1) $ DUMP(Z,2,1) $
    TIME=T $
   'END' $
    X0=X0+XDOT*TIME $ Y0=Y0+YDOT*TIME $
    XDOT=XDOT+CHANGE OF XDOT $
    YDOT=YDOT+CHANGE OF YDOT $
    FINISH=FINISH-TIME $
    'FOR'T=DELTAT'STEP'DELTAT 'UNTIL' FINISH 'DO'
    'BEGIN'
      EX=X0+XDOT*T $ WY=Y0+YDOT*T $
      'IF' NOISE 'EQ' 1 'THEN' NOISEPLOT $
      'IF' NOISE 'EQ' 0 'THEN' CLEANPLOT $
      PHI(1,2)=PHI(3,4)=DELTAT $
      TRACK(2,4,PHI,M,MT,P,X,Z,R) $
      DUMP(X,4,1) $ DUMP(Z,2,1) $
    'END' $
 'END'
```

APPENDIX F

POLAR FIRST ORDER TRACKER - PROGRAM HARD COPY

```
*IODUNIT10/PRINT
*IODUNIT20/READ
*XEQ
*STORAGE/4000/15000
*CHAIN1
*ALGOL
   'BEGIN INTEGER' I,J,MEASURE,STATE,MI,N,O,OLDX¢
     'ARRAY' X(1:4,1:1), R(1:2,1:2), Z(1:2,1:1), COPYR(1:2,1:2),
     COPYZ(1:2,1:1), P(1:4,1:4), M(1:2,1:4),MT(1:4,1:2),
     PHI(1:4,1:4), K(1:4,1:2),PMT(1:4,1:2),OUTX(1:4,1:1),
     OUTZ(1:2,1:1) ¢
     'REAL' T,TIME,DELTAT,CHANGE,FINISH,FX,XDOT,CHANGE OF XDOT,
     WY,YDOT,CHANGE OF YDOT,X0,Y0,VARR,VARTHETA,RNOISE,THETANOISE,
     U1,U2,VARIANCE,NOISE¢

     'COMMENT' X,Z AND COPYZ ARE COLUMN VECTORS.
     R AND COPYR ARE 2*2 MEASUREMENT COVARIANCE MATRICES.
     K IS A 4*2 MATRIX, AND IS ESSENTIALLY THE DAMPING FACTOR.
     M IS THE MEASUREMENT MATRIX AND IS 2*4.
     MT IS THE TRANSPOSE OF M.
     TRACK IN R,THETA,GENERATED IN X,Y WITH CONSTANT DATA RATE.
     MANOEUVRE AT TIME CHANGE. FINISH AT TIME FINISH.
     KALMAN FILTER TRACKER WITHOUT PLANT NOISE TERMS ¢

     'PROCEDURE' TRANSPOSE(PHI,PHIT,MI,N) ¢
     'INTEGER' MI,N ¢
     'ARRAY' PHI,PHIT ¢
     'BEGIN INTEGER' I,J ¢
        'ARRAY' COPYPHI(1:MI,1:N) ¢
        'FOR'I=1'STEP'1'UNTIL'MI'DO'
        'FOR'J=1'STEP'1'UNTIL'N'DO'
        COPYPHI(I,J)=PHI(I,J) ¢
        'COMMENT' PHI COPIED SO THAT TRANSPOSE CAN BE ASSIGNED
        TO ORIGINAL MATRIX(ONLY POSSIBLE FOR SQUARE MATRICES) ¢
        'FOR'I=1'STEP'1'UNTIL'MI'DO'
        'FOR'J=1'STEP'1'UNTIL'N'DO'
        PHIT(J,I)=COPYPHI(I,J) ¢
     'END'TRANSPOSE ¢

     'PROCEDURE' MATRIXINV(A,B,MI) ¢
     'ARRAY' A,B ¢
     'INTEGER' MI ¢
     'BEGIN REAL' DET ¢
     'COMMENT' TO INVERT A 2*2 MATRIX ¢
        'IF'MI'NE'2'THEN BEGIN'
           WRITETC(10,'(('C')'M NE 2')') ¢
           'GOTO' STOP ¢
        'END' ¢
        DET=A(1,1)*A(2,2)-A(1,2)*A(2,1) ¢
        B(1,1)=A(2,2)/DET ¢
        B(1,2)=(-A(1,2)/DET) ¢
        B(2,1)=(-A(2,1)/DET) ¢
        B(2,2)=A(1,1)/DET ¢
     STOP:'END' MATRIXINV ¢
```

```
'PRJCEDURE' MATRIXADD(A,B,C,MI,N) ¢
'ARRAY' A,B,C ¢
'INTEGER' MI,N ¢
'CJMMENT' THE SUM JF THE MATRICES IS STJRED IN THE ARRAY
C.A,B,C ARE MI*N ¢
'BEGIN INTEGER' I,J,P ¢
   'FJR'I=1'STEP'1'UNTIL'MI'DJ'
   'FJR'J=1'STEP'1'UNTIL'N'DJ'
   C(I,J)=A(I,J)+B(I,J) ¢
'END' MATRIXADD ¢

'PRJCEDURE' MATRIXSUB(A,B,C,MI,N) ¢
'ARRAY' A,B,C ¢
'INTEGER' MI,N ¢
'CJMMENT' THE DIFFERENCE JF THE TWJ MATRICES IS STJRED IN THE
ARRAY C. A,B,C ARE MI*N ¢
'BEGIN INTEGER' I,J ¢
   'FJR'I=1'STEP'1'UNTIL'MI'DJ'
   'FJR'J=1'STEP'1'UNTIL'N'DJ'
   C(I,J)=A(I,J)-B(I,J) ¢
'END' MATRIXSUB ¢

'PRJCEDURE' MATRIXMULT(A,B,C,MI,N,P) ¢
'ARRAY' A,B,C ¢
'INTEGER' MI,N,P ¢
'CJMMENT' THE PRJDUCT JF THE TWJ ARRAYS IS STJRED IN THE
ARRAY C. TJ MULTIPLY BY A SCALAR THE SECJND
MATRIX MUST HAVE THE VALUE JF THE SCALAR AS ITS DIAGJNAL
WITH THE REST JF THE ELEMENTS AS NJUGHTS. A IS M*N,B IS N*P,
C IS M*P. CJPIES JF INPUT ARRAYS ARE MADE IN JRDER
THAT AN ARRAY MAY BE USED BJTH AS AN INPUT AND
JUTPUT PARAMETER ¢
'BEGIN INTEGER' I,J,X ¢
   'ARRAY' CJPYA(1:MI,1:N),CJPYB(1:N,1:P) ¢
   'FJR'I=1'STEP'1'UNTIL'MI'DJ'
   'FJR'J=1'STEP'1'UNTIL'N'DJ'
   CJPYA(I,J)=A(I,J) ¢
   'FJR'I=1'STEP'1'UNTIL'N'DJ'
   'FJR'J=1'STEP'1'UNTIL'P'DJ'
   CJPYB(I,J)=B(I,J) ¢
   'FJR'I=1'STEP'1'UNTIL'MI'DJ'
   'FJR'J=1'STEP'1'UNTIL'P'DJ'
   'BEGIN' C(I,J)=0 ¢
      'FJR'X=1'STEP'1'UNTIL'N'DJ'
      C(I,J)=CJPYA(I,X)*CJPYB(X,J)+C(I,J) ¢
   'END' ¢
'END' MATRIXMULT ¢
```

```
'PROCEDURE' DUMP(A,MI,N) ¢
'ARRAY' A ¢
'INTEGER' MI,N ¢
'BEGIN 'INTEGER'I,J ¢
   'FOR'I=1 'STEP'1 'UNTIL'MI 'DO'
   'BEGIN'
      'FOR'J=1 'STEP'1 'UNTIL'N 'DO'
      WRITE(10,LAYOUT('('S-NDDDDD.DDD')'),A(I,J)) ¢
   'END' ¢
   NEWLIN(10,4) ¢
'END' DUMP ¢

'PROCEDURE' CARTCONV ¢
'BEGIN'
   OUTX(1,1)=X(1,1)*SIN(X(3,1)) ¢
   OUTX(2,1)=X(2,1)*SIN(X(3,1))+X(1,1)*X(4,1)*COS(X(3,1)) ¢
   OUTX(3,1)=X(1,1)*COS(X(3,1)) ¢
   OUTX(4,1)=X(2,1)*COS(X(3,1))-X(1,1)*X(4,1)*
   SIN(X(3,1)) ¢
   OUTZ(1,1)=Z(1,1)*SIN(Z(2,1)) ¢
   OUTZ(2,1)=Z(1,1)*COS(Z(2,1)) ¢
'END' CARTCONV ¢

'REAL PROCEDURE' RANDOM ¢
'BEGIN INTEGER' CALC ¢
'COMMENT' OLDX MUST BE DECLARED GLOBALLY ¢
   OLDX=255*OLDX ¢
   CALC=ENTIER(OLDX/131071) ¢
   OLDX=OLDX-131071*CALC ¢
   RANDOM=OLDX/131071 ¢
'END' RANDOM ¢

'PROCEDURE' GAUSSIAN(NOISE,VARIANCE) ¢
'REAL 'NOISE,VARIANCE¢
'BEGIN REAL' TWOPI ¢
   TWOPI=6.28318 ¢
   U1=RANDOM ¢
   U2=RANDOM ¢
   NOISE=SQRT(-2*VARIANCE* LN(U1))*COS(TWOPI*U2) ¢
'END' GAUSSIAN ¢

'PROCEDURE' SUBST ¢
'BEGIN COMMENT' TO SET Z MATRIX WHEN NOISE ADDED ¢
   THETANOISE=RNOISE=0 ¢
   GAUSSIAN(RNOISE,VARR) ¢
   GAUSSIAN(THETANOISE,VARTHETA) ¢
   Z(1,1)=SQRT(EX*EX+WY*WY)+RNOISE ¢
   Z(2,1)=ARCTAN(EX/WY)+THETANOISE ¢
'END' SUBST ¢
```

```
/PROCEDURE/ TRACK(MEASURE,STATE,PHI,M,MT,P,X,Z,R) ¢
/ARRAY/ PHI,P,X,Z,R,MT,M ¢
/INTEGER/ MEASURE,STATE ¢
/BEGIN ARRAY/ MP(1:MEASURE,1:STATE),PHIT(1:STATE,1:STATE),
MPMT(1:MEASURE,1:MEASURE),PREDZ(1:MEASURE,1:1),
ERRORCOV(1:MEASURE,1:MEASURE),CORRECTION(1:STATE,1:1),
KMP(1:STATE,1:STATE) ¢
   TRANSPOSE(PHI,PHIT,4,4) ¢
   MATRIXMULT(PHI,X,X,4,4,1) ¢
   MATRIXMULT(P,PHIT,P,4,4,4) ¢
   MATRIXMULT(PHI,P,P,4,4,4) ¢
   MATRIXMULT(M,P,MP,2,4,4) ¢
   MATRIXMULT(P,MT,PMT,4,4,2) ¢
   MATRIXMULT(M,PMT,MPMT,2,4,2) ¢
   MATRIXADD(MPMT,R,MPMT,2,2) ¢
   MATRIXINV(MPMT,ERRORCOV,2) ¢
   MATRIXMULT(PMT,ERRORCOV,K,4,2,2) ¢
   MATRIXMULT(M,X,PREDZ,2,4,1) ¢
   MATRIXSUB(PREDZ,Z,PREDZ,2,1) ¢
   MATRIXMULT(K,PREDZ,CORRECTION,4,2,1) ¢

   /COMMENT/ PREDZ IS NOW (M*X-Z)*K ¢

   MATRIXSUB(X,CORRECTION,X,4,1) ¢
   MATRIXMULT(K,MP,KMP,4,2,4) ¢
   MATRIXSUB(P,KMP,P,4,4) ¢

   /COMMENT/ FOR A 4 VARIABLE STATE VECTOR(STATE=4),
   AND A TWO VARIABLE MEASUREMENT VECTOR(MEASURE=2),
   X IS 4*1,Z IS 2*1, P IS 4*4, M IS 2*4, PHI IS 4*4,K IS 4*2 ¢
/END/ TRACK ¢

M(1,1)=M(2,3)=MT(1,1)=MT(3,2)=1 ¢
M(1,2)=M(1,3)=M(1,4)=M(2,1)=M(2,2)=M(2,4)=MT(1,2)=MT(2,2)=MT(2,1)=
MT(3,1)=MT(4,1)=MT(4,2)=0 ¢
/FOR/I=1/STEP/1/UNTIL/4/DO/
/FOR/J=1/STEP/1/UNTIL/4/DO/
/BEGIN/
   PHI(I,J)=/IF/I/EQ/J/THEN/1/ELSE/0 ¢
   P(I,J)=0 ¢
/END/ ¢
STATE=4 ¢  MEASURE=2 ¢

/COMMENT/ STATE VECTOR IS 4-VARIABLE,MEASUREMENT VECTOR
IS 2-VARIABLE ¢

X0=READ(20) ¢  Y0=READ(20) ¢
XDOT=READ(20) ¢  YDOT=READ(20) ¢
CHANGE OF XDOT=READ(20) ¢  CHANGE OF YDOT=READ(20) ¢
CHANGE=READ(20) ¢  FINISH=READ(20) ¢  DELTAT=READ(20) ¢
VARR=READ(20) ¢  VARTHETA=READ(20) ¢  OLDX=READ(20) ¢
NOISE=READ(20) ¢
```

```
    WRITET(10, '('/C')'1ST'*'ORDER'*'KALMAN'('/C'))') ;
INITIATE TRACK: 'IF' NOISE 'EQ' 1 'THEN'
    'BEGIN'
       GAUSSIAN(RNOISE, VARR) ;
         GAUSSIAN(THETANOISE, VARTHETA) ;
    'END' 'ELSE' RNOISE=THETANOISE=0 ;
    COPYZ(1,1)=SQRT(X0*X0+Y0*Y0)+RNOISE ;
    COPYZ(2,1)=ARCTAN(X0/Y0)+THETANOISE ;
    CARTCONV ;
    DUMP(OUTZ,2,1) ;
    COPYR(1,1)=R(1,1)=VARR ;
    COPYR(1,2)=R(1,2)=COPYR(2,1)=R(2,1)=0 ;
    COPYR(2,2)=R(2,2)=VARTHETA ;
    T=DELTAT ;
    EX=X0+XDOT*DELTAT ;
    WY=Y0+YDOT*DELTAT ;
    'IF'NOISE'EQ'1'THEN' SUBST ;
    'IF'NOISE'EQ'0'THEN'
    'BEGIN'
       Z(1,1)=SQRT(EX*EX+WY*WY) ;
       Z(2,1)=ARCTAN(EX/WY) ;
    'END' ;
    TIME=T ;
    X(1,1)=Z(1,1) ;
    X(2,1)=(Z(1,1)-COPYZ(1,1))/DELTAT ;
    X(3,1)=Z(2,1) ;
    X(4,1)=(Z(2,1)-COPYZ(2,1))/DELTAT ;
    P(1,1)=R(1,1) ;
    P(1,2)=P(2,1)=R(1,1)/DELTAT ;
    P(2,2)=(R(1,1)+COPYR(1,1))/(DELTAT*DELTAT) ;
    P(3,3)=R(2,2) ;
    P(3,4)=P(4,3)=R(2,2)/DELTAT ;
    P(4,4)=(R(2,2)+COPYR(2,2))/(DELTAT*DELTAT) ;

    'COMMENT' TRACK IS NOW INITIATED ;

    CARTCONV ;
       DUMP(OUTX,4,1) ;
       DUMP(OUTZ,2,1) ;
    'FOR'T=2*DELTAT'STEP'DELTAT'UNTIL'CHANGE'DO'
    'BEGIN'
       EX=X0+XDOT*T ;
       WY=Y0+YDOT*T ;
       'IF'NOISE'EQ'1'THEN' SUBST ;
        'IF'NOISE'EQ'0'THEN'
       'BEGIN'
         Z(1,1)=SQRT(EX*EX+WY*WY) ;
           Z(2,1)=ARCTAN(EX/WY) ;
       'END' ;

       'COMMENT' NOISE ADDED,WITH VARIANCE VARR AND VARTHETA
       IN THE TWO PRECEDING STATEMENTS ;
```

```
        PHI(1,2)=PHI(3,4)=DELTAT ;
        TRACK(2,4,PHI,M,MT,P,X,Z,R) ;
        CARTCONV ;
        DUMP(OUTX,4,1) ;
        DUMP(OUTZ,2,1) ;
        TIME=T ;
    'END' ;
    X0=X0+XDOT*TIME ;
    Y0=Y0+YDOT*TIME ;
    XDOT=XDOT+CHANGE OF XDOT ;
    YDOT=YDOT+CHANGE OF YDOT ;
    FINISH=FINISH-TIME ;
    'FOR'T=DELTAT'STEP'DELTAT'UNTIL'FINISH'DO'
    'BEGIN'
        EX=X0+XDOT*T ;
        WY=Y0+YDOT*T ;
         'IF'NOISE'EQ'1'THEN'SUBST ;
        'IF'NOISE'EQ'0'THEN'
        'BEGIN'
            Z(1,1)=SQRT(EX*EX+WY*WY) ;
            Z(2,1)=ARCTAN(EX/WY) ;
        'END' ;

        'COMMENT' NOISE ADDED, WITH VARR AND VARTHETA IN THE
        TWO PRECEDING STATEMENTS ;

        PHI(1,2)=PHI(3,4)=DELTAT ;
        TRACK(2,4,PHI,M,MT,P,X,Z,R) ;
        CARTCONV ;
        DUMP(OUTX,4,1) ;
        DUMP(OUTZ,2,1) ;
    'END' ;
'END'
```

APPENDIX G

POLAR SECOND ORDER TRACKER - PROGRAM HARD COPY

```
*IODUNIT10/PRINT
*IODUNIT20/READ
*XEQ
*STORAGE/4000/15000
*CHAIN1
*ALGOL
  'BEGIN INTEGER' I,J,MEASURE,STATE,OLDX,F$
    'ARRAY' X(1:6,1:1),Z(1:2,1:1),OLDZ(1:2,1:1),
    OLDESTZ(1:2,1:1),PHI(1:6,1:6),P(1:6,1:6),
    R(1:2,1:2),OLDR(1:2,1:2),OLDESTR(1:2,1:2),
    M(1:2,1:6),MT(1:6,1:2),OUTX(1:4,1:1),OUTZ(1:2,1:1) $
    'REAL' T,TIME,DELTAT,CHANGE,FINISH,EX,XDOT,CHANGE OF XDOT,
    WY,YDOT,CHANGE OF YDOT,X0,Y0,VARR,VARTHETA,DELTATSQ,
    RNOISE,THETANOISE,U1,U2,VARIANCE,NOISE $

    'COMMENT' X,Z AND OLDZ ARE COLUMN VECTORS.
    R AND OLDR ARE 2*2 MEASUREMENT COVARIANCE MATRICES.
    K IS A 6*2 MATRIX, AND IS ESSENTIALLY THE DAMPING FACTOR.
    M IS THE MEASUREMENT MATRIX AND IS 2*6.
    MT IS THE TRANSPOSE OF M.
    TRACK IN R,THETA,RDOT,THETADOT,RDOUBLEDOT,THETADOUBLEDOT,
    GENERATED IN X,Y WITH CONSTANT DATA RATE.
    MANOEUVRE AT TIME CHANGE. FINISH AT TIME FINISH.
    KALMAN FILTER TRACKER WITHOUT PLANT NOISE TERMS $

    'PROCEDURE' TRANSPOSE(PHI,PHIT,MI,N) $
    'INTEGER' MI,N $
    'ARRAY' PHI,PHIT $
    'BEGIN INTEGER' I,J $
       'ARRAY' COPYPHI(1:MI,1:N) $
       'FOR'I=1'STEP'1'UNTIL'MI'DO'
       'FOR'J=1'STEP'1'UNTIL'N'DO'
       COPYPHI(I,J)=PHI(I,J) $
       'COMMENT' PHI COPIED SO THAT TRANSPOSE CAN BE ASSIGNED
       TO ORIGINAL MATRIX(ONLY POSSIBLE FOR SQUARE MATRICES) $
       'FOR'I=1'STEP'1'UNTIL'MI'DO'
       'FOR'J=1'STEP'1'UNTIL'N'DO'
       PHIT(J,I)=PHI(I,J) $
    'END'TRANSPOSE $

    'PROCEDURE' MATRIXINV(A,B,MI) $
    'ARRAY' A,B $
    'INTEGER' MI $
    'BEGIN REAL' DET $
    'COMMENT' TO INVERT A 2*2 MATRIX $
       'IF'MI'NE'2'THEN BEGIN'
          WRITET(10,'('('C')'M NE 2')') $
          'GOTO' STOP $
       'END' $
       DET=A(1,1)*A(2,2)-A(1,2)*A(2,1) $
       B(1,1)=A(2,2)/DET $
       B(1,2)=(-A(1,2)/DET) $
       B(2,1)=(-A(2,1)/DET) $
       B(2,2)=A(1,1)/DET $
    STOP:'END' MATRIXINV $
```

```
'PROCEDURE' MATRIXADD(A,B,C,MI,N) $
'ARRAY' A,B,C $
'INTEGER' MI,N $
'COMMENT' THE SUM OF THE MATRICES IS STORED IN THE ARRAY
C.A,B,C ARE MI*N $
'BEGIN INTEGER' I,J,P $
   'FOR'I=1 'STEP'1 'UNTIL 'MI 'DO'
   'FOR'J=1 'STEP'1 'UNTIL 'N 'DO'
   C(I,J)=A(I,J)+B(I,J) $
'END' MATRIXADD $

'PROCEDURE' MATRIXSUB(A,B,C,MI,N) $
'ARRAY' A,B,C $
'INTEGER' MI,N $
'COMMENT  THE DIFFERENE OF THE TWO MATRICES IS STORED IN THE
ARRAY C. A,B,C ARE MI*N $
'BEGIN INTEGER' I,J $
   'FOR'I=1 'STEP'1 'UNTIL 'MI 'DO'
   'FOR'J=1 'STEP'1 'UNTIL 'N 'DO'
   C(I,J)=A(I,J)-B(I,J) $
'END' MATRIXSUB $

'PROCEDURE' MATRIXMULT(A,B,C,MI,N,P) $
'ARRAY' A,B,C $
'INTEGER' MI,N,P $
'COMMENT' THE PRODUCT OF THE TWO ARRAYS IS STORED IN THE
ARRAY C. TO MULTIPLY BY A SCALAR THE SECOND
MATRIX MUST HAVE THE VALUE OF THE SCALAR AS ITS DIAGONAL
WITH THE REST OF THE ELEMENTS AS NOUGHTS. A IS M*N,B IS N*P,
C IS M*P. COPIES OF INPUT ARRAYS ARE MADE IN ORDER
THAT AN ARRAY MAY BE USED BOTH AS AN INPUT AND
OUTPUT PARAMETER $
'BEGIN INTEGER' I,J,X $
   'ARRAY' COPYA(1:MI,1:N),COPYB(1:N,1:P) $
   'FOR'I=1 'STEP'1 'UNTIL 'MI 'DO'
   'FOR'J=1 'STEP'1 'UNTIL 'N 'DO'
   COPYA(I,J)=A(I,J) $
   'FOR'I=1 'STEP'1 'UNTIL 'N 'DO'
   'FOR'J=1 'STEP'1 'UNTIL 'P 'DO'
   COPYB(I,J)=B(I,J) $
   'FOR'I=1 'STEP'1 'UNTIL 'MI 'DO'
   'FOR'J=1 'STEP'1 'UNTIL 'P 'DO'
   'BEGIN' C(I,J)=0 $
      'FOR'X=1 'STEP'1 'UNTIL 'N 'DO'
      C(I,J)=COPYA(I,X)*COPYB(X,J)+C(I,J) $
   'END' $
'END' MATRIXMULT $
```

```
'PROCEDURE' DUMP(A,MI,N) $
'ARRAY' A $
'INTEGER' MI,N $
'BEGIN INTEGER'I,J $
   'FOR'I=1'STEP'1'UNTIL'MI'DO'
   'BEGIN'
      'FOR'J=1'STEP'1'UNTIL'N'DO'
      WRITE(10,LAYOUT('('S-NDDDDD.DDD')'),A(I,J)) $
   'END' $
   NEWLIN(10,4) $
'END' DUMP $

'PROCEDURE' CONVERT $
'BEGIN'
   OUTX(1,1)=X(1,1)*SIN(X(4,1)) $
   OUTX(2,1)=X(2,1)*SIN(X(4,1))+X(1,1)*X(5,1)
   *COS(X(4,1)) $
   OUTX(3,1)=X(1,1)*COS(X(4,1)) $
   OUTX(4,1)=X(2,1)*COS(X(4,1))-X(1,1)*X(5,1)
   *SIN(X(4,1)) $
   OUTZ(1,1)=Z(1,1)*SIN(Z(2,1)) $
   OUTZ(2,1)=Z(1,1)*COS(Z(2,1)) $
'END' CONVERT $

'REAL PROCEDURE' RANDOM $
'BEGIN INTEGER' CALC $
'COMMENT' OLDX MUST BE DECLARED GLOBALLY $
OLDX=255*OLDX $
CALC=ENTIER(OLDX/131071) $
OLDX=OLDX-131071*CALC $
RANDOM=OLDX/131071 $
'END' RANDOM $

'PROCEDURE' GAUSSIAN(NOISE,VARIANCE) $
'REAL' NOISE,VARIANCE $
'BEGIN REAL' TWOPI $
   TWOPI=6.28318 $
   U1=RANDOM $
   U2=RANDOM $
   NOISE=SQRT(-2*VARIANCE*LN(U1))*COS(TWOPI*U2) $
'END' GAUSSIAN $

'PROCEDURE' SUBSTITUTE $
'BEGIN COMMENT' TO SET Z MATRIX WHEN NOISE ADDED $
   THETANOISE=RNOISE=0 $
   GAUSSIAN(RNOISE,VARR) $
   GAUSSIAN(THETANOISE,VARTHETA) $
   Z(1,1)=SQRT(EX*EX+WY*WY)+RNOISE$
   Z(2,1)=ARCTAN(EX/WY)+THETANOISE $
'END' SUBSTITUTE $
```

```
'PROCEDURE' TRACK(MEASURE, STATE, PHI, M, MT, P, X, Z, R) $
'ARRAY' PHI, P, X, Z, R, MT, M $
'INTEGER' MEASURE, STATE $
'BEGIN ARRAY' MP(1:MEASURE, 1:STATE), PHIT(1:STATE, 1:STATE),
MPMT(1:MEASURE, 1:MEASURE), PREDZ(1:MEASURE, 1:1),
ERRORCOV(1:MEASURE, 1:MEASURE), CORRECTION(1:STATE, 1:1),
KMP(1:STATE, 1:STATE), PMT(1:STATE, 1:MEASURE), K(1:STATE, 1:MEASURE) $
   TRANSPOSE(PHI, PHIT, STATE, STATE) $
   MATRIXMULT(PHI, X, X, STATE, STATE, 1) $
   MATRIXMULT(P, PHIT, P, STATE, STATE, STATE) $
   MATRIXMULT(PHI, P, P, STATE, STATE, STATE) $
   MATRIXMULT(M, P, MP, MEASURE, STATE, STATE) $
   MATRIXMULT(P, MT, PMT, STATE, STATE, MEASURE) $
   MATRIXMULT(M, PMT, MPMT, MEASURE, STATE, MEASURE) $
   MATRIXADD(MPMT, R, MPMT, MEASURE, MEASURE) $
   MATRIXINV(MPMT, ERRORCOV, MEASURE) $
   MATRIXMULT(PMT, ERRORCOV, K, STATE, MEASURE, MEASURE) $
   MATRIXMULT(M, X, PREDZ, MEASURE, STATE, 1) $
   MATRIXSUB(PREDZ, Z, PREDZ, MEASURE, 1) $
   MATRIXMULT(K, PREDZ, CORRECTION, STATE, MEASURE, 1) $

    'COMMENT' PREDZ IS NOW (M*X-Z)*K $

   MATRIXSUB(X, CORRECTION, X, STATE, 1) $
   MATRIXMULT(K, MP, KMP, STATE, MEASURE, STATE) $
   MATRIXSUB(P, KMP, P, STATE, STATE) $

    'COMMENT' FOR A 6 VARIABLE STATE VECTOR(STATE=6),
    AND A TWO VARIABLE MEASUREMENT VECTOR(MEASURE=2),
    X IS 6*1, Z IS 2*1, P IS 6*6, M IS 2*6, PHI IS 6*6, K IS 6*2 $
 'END' TRACK $

 'COMMENT' SETTING OF MEASUREMENT MATRIX $
 'FOR'I=1,2'DO'
 'FOR'J=1'STEP'1'UNTIL'6'DO'
 M(I,J)=MT(J,I)=0 $
 M(1,1)=MT(1,1)=M(2,4)=MT(4,2)=1 $
 'COMMENT' SETTING TRANSITIONAL MATRIX AND STATE CONVARIANCE
 MATRIX $
 'FOR'I=1'STEP'1'UNTIL'6'DO'
 'FOR'J=1'STEP'1'UNTIL'6'DO'
 'BEGIN'
    PHI(I,J)='IF'I'EQ'J'THEN'1'ELSE'0 $
    P(I,J)=0 $
 'END' $
 STATE=6 $  MEASURE=2 $

 'COMMENT' STATE VECTOR IS 6-VARIABLE,MEASUREMENT VECTOR
 IS 2-VARIABLE $

 X0=READ(20) $  Y0=READ(20) $
 XDOT=READ(20) $  YDOT=READ(20) $
 CHANGE OF XDOT=READ(20) $  CHANGE OF YDOT=READ(20) $
 CHANGE=READ(20) $  FINISH=READ(20) $  DELTAT=READ(20) $
 VARR=READ(20) $  VARTHETA=READ(20) $  OLDX=READ(20) $
 DELTATSQ=DELTAT*DELTAT $ NOISE=READ(20) $
```

```
   WRITETC(10, '(( 'C') '2ND'* 'ORDER'* 'KALMAN '( 'C')) ') $
IN TIATE TRACK: 'IF' NOISE 'EQ' 1 'THEN'
   'BEGIN'
     GAUSSIAN(RNOISE, VARR) $
      GAUSSIAN(THETANOISE, VARTHETA) $
   'END' 'ELSE' RNOISE=THETANOISE=0 $
   OLDESTZ(1, 1)=SQRT(X0*X0+Y0*Y0)+RNOISE $
   OLDESTZ(2, 1)=ARCTAN(X0/Y0)+THETANOISE $
   CONVERT $
   DUMP(OUTZ, 2, 1) $
   F=LAYOUT( '( 'S-NDDDD.DDDD') ') $
   WRITE(10,LAYOUT( '( 'S-NDD.DDD') '), X0) $
   NEWLIN(10, 1) $
   WRITE(10,LAYOUT( '( 'S-NDD.DDD') '), Y0) $
   NEWLIN(10, 4) $
   EX=X0+XDOT*DELTAT $
   WY=Y0+YDOT*DELTAT $
   'IF'NOISE 'EQ' 1 'THEN'
   'BEGIN'
      GAUSSIAN(RNOISE, VARR) $
      GAUSSIAN(THETANOISE, VARTHETA) $
   'END' 'ELSE' RNOISE=THETANOISE=0 $
   OLDZ(1, 1)=SQRT(EX*EX+WY*WY)+RNOISE $
   OLDZ(2, 1)=ARCTAN(EX/WY)+THETANOISE $
   EX=EX+XDOT*DELTAT $
   WY=WY+YDOT*DELTAT $
   SUBSTITUTE $
   R(1, 1)=OLDR(1, 1)=OLDESTR(1, 1)=VARR $
   R(2, 2)=OLDR(2, 2)=OLDESTR(2, 2)=VARTHETA $
   R(1, 2)=R(2, 1)=OLDR(1, 2)=OLDR(2, 1)=OLDESTR(1, 2)=OLDESTR(2, 1)=0 $
INITIATION:X(1, 1)=Z(1, 1) $
   X(2, 1)=(3*Z(1, 1)-4*OLDZ(1, 1)+OLDESTZ(1, 1))/(2*DELTAT) $
   X(3, 1)=(OLDESTZ(1, 1)-2*OLDZ(1, 1)+Z(1, 1))/DELTATSQ $
   X(4, 1)=Z(2, 1) $
   X(5, 1)=(3*Z(2, 1)-4*OLDZ(2, 1)+OLDESTZ(2, 1))/(2*DELTAT) $
   X(6, 1)=(OLDESTZ(2, 1)-2*OLDZ(2, 1)+Z(2, 1))/DELTATSQ $
   'FOR'I=1, 4'DO'
   'BEGIN'
      J='IF'I'EQ'1'THEN'1'ELSE'2 $
   P(I, I)=R(J, J) $
   P(I, I+1)=P(I+1, I)=3*R(J, J)/(DELTAT*2) $
   P(I, I+2)=P(I+2, I)=R(J, J)/DELTATSQ $
   P(I+1, I+1)=(OLDESTR(J, J)+16*OLDR(J, J)+9*R(J, J))/
   (4*DELTATSQ) $
   P(I+2, I+1)=P(I+1, I+2)=(OLDESTR(J, J)+8*OLDR(J, J)
   +3*R(J, J))/(2*DELTATSQ*DELTAT) $
   P(I+2, I+2)=(OLDESTR(J, J)+4*OLDR(J, J)+R(J, J))/
   (DELTATSQ*DELTATSQ) $
   'END' $
      CONVERT $
      DUMP(OUTX, 4, 1) $
      DUMP(OUTZ, 2, 1) $
   'COMMENT' TRACK IS NOW INITIATED: TAKES 3 PLOTS
   TO INITIATE WITH A 2ND ORDER TRACKER $
```

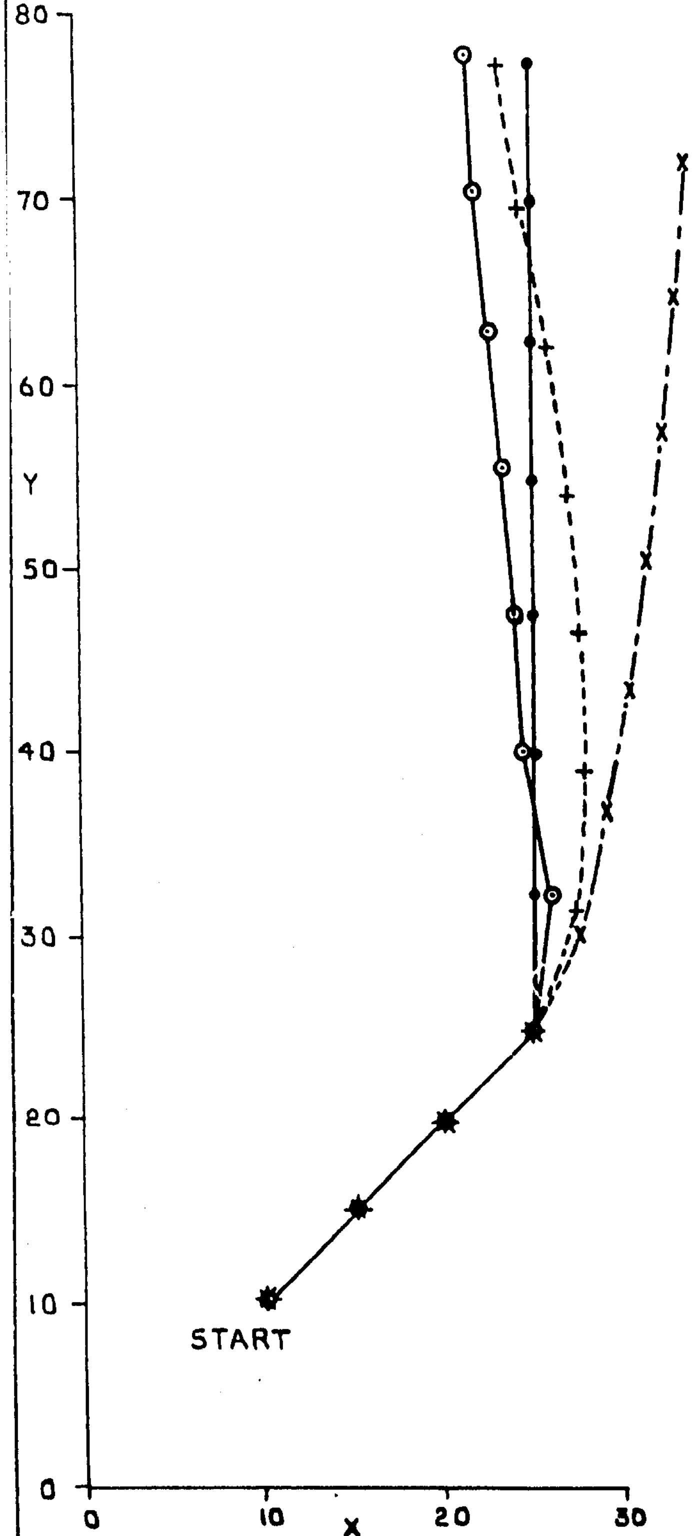
```
GENERATION: 'FOR'T=3*DELTAT'STEP'DELTAT'UNTIL'CHANGE'DO'
   'BEGIN'
      EX=X0+XDOT*T $
      WY=Y0+YDOT*T $
      SUBSTITUTE $
      PHI(1,2)=PHI(2,3)=PHI(4,5)=PHI(5,6)=DELTAT $
      PHI(1,3)=PHI(4,6)=DELTATSQ/2 $
      TRACK(2,6,PHI,M,MT,P,X,Z,R) $
      CONVERT $
      DUMP(OUTX,4,1) $
      DUMP(OUTZ,2,1) $
      TIME=T $
   'END' $
   X0=X0+XDOT*TIME $
   Y0=Y0+YDOT*TIME $
   XDOT=XDOT+CHANGE OF XDOT $
   YDOT=YDOT+CHANGE OF YDOT $
   FINISH=FINISH-TIME $
   'FOR'T=DELTAT'STEP'DELTAT'UNTIL'FINISH'DO'
   'BEGIN'
      EX=X0+XDOT*T $
      WY=Y0+YDOT*T $
      SUBSTITUTE $
      PHI(1,2)=PHI(2,3)=PHI(4,5)=PHI(5,6)=DELTAT $
      PHI(1,3)=PHI(4,6)=DELTATSQ/2 $
      TRACK(2,6,PHI,M,MT,P,X,Z,R) $
      CONVERT $
      DUMP(OUTX,4,1) $
      DUMP(OUTZ,2,1) $
   'END' $
'END'
```

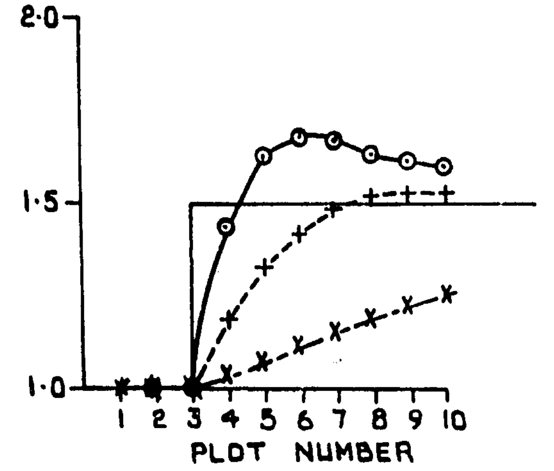
RESPONSE TO A TURN WITH NOISE FREE DATA

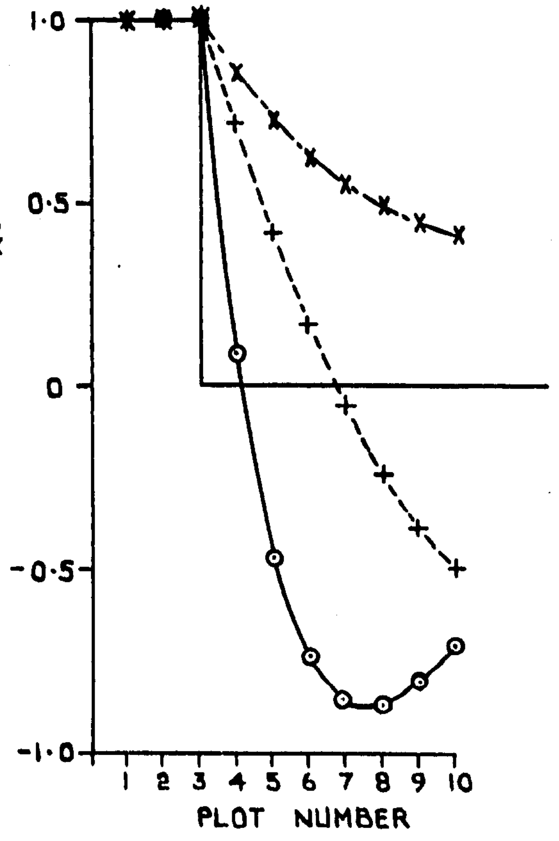
1

- ● TRUE POSITION, VELOCITY
- + POSITION OR VELOCITY FROM 1st ORDER R-θ TRACKER
- ⊙ POSITION OR VELOCITY FROM 2nd ORDER R-θ TRACKER
- X POSITION OR VELOCITY FROM X-Y TRACKER

DATA INTERVALS 5 TIME UNITS







NOTE Ẋ AND Ẏ ARE DISTANCE UNITS / TIME UNIT

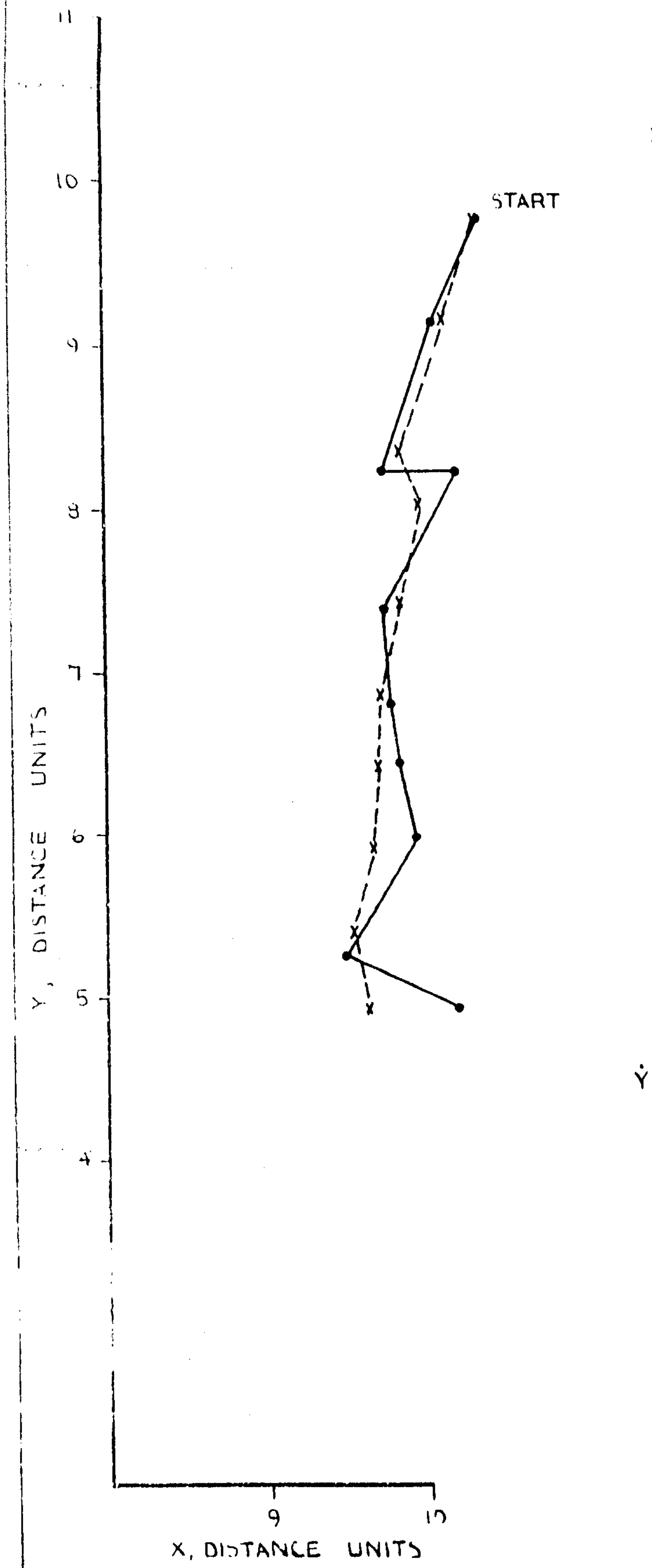
A.S.W.E. DATE TR. CH. APR. DRG No

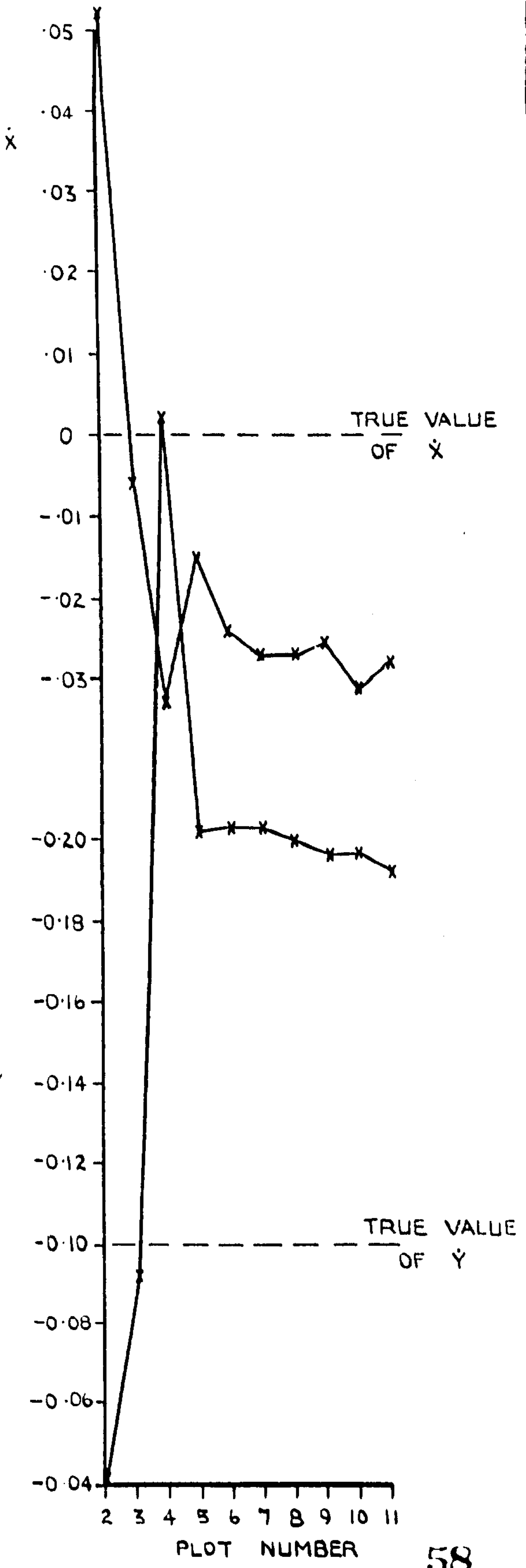
1st ORDER KALMAN $(R, \theta, \dot{R}, \dot{\theta})$ $\dot{X} = 0$, $\dot{Y} = 0{\cdot}1$ d.u./t.u.

2

—●— = INPUT DATA
---x--- = SMOOTHED OUTPUT

DATA INTERVAL = 5 TIME UNITS
SPEED = 0·1 DU/TU
CROSSING DISTANCE = 10





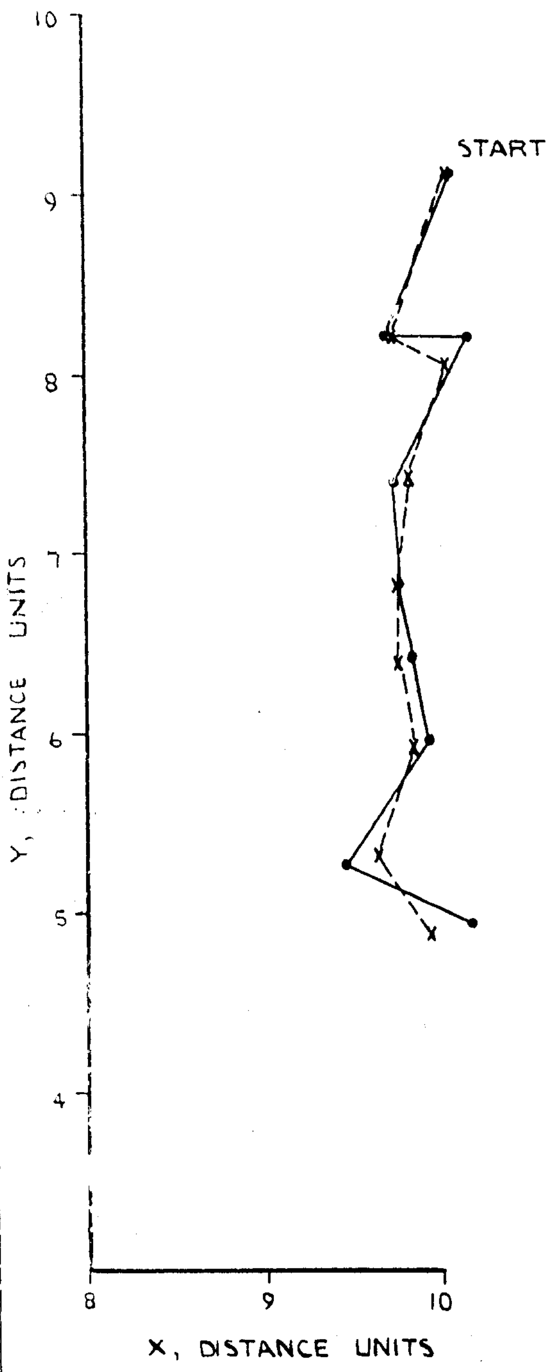
A.S.W.E. DATE 20.4.72. TR. C. HARRISON CH APR DRG. No.

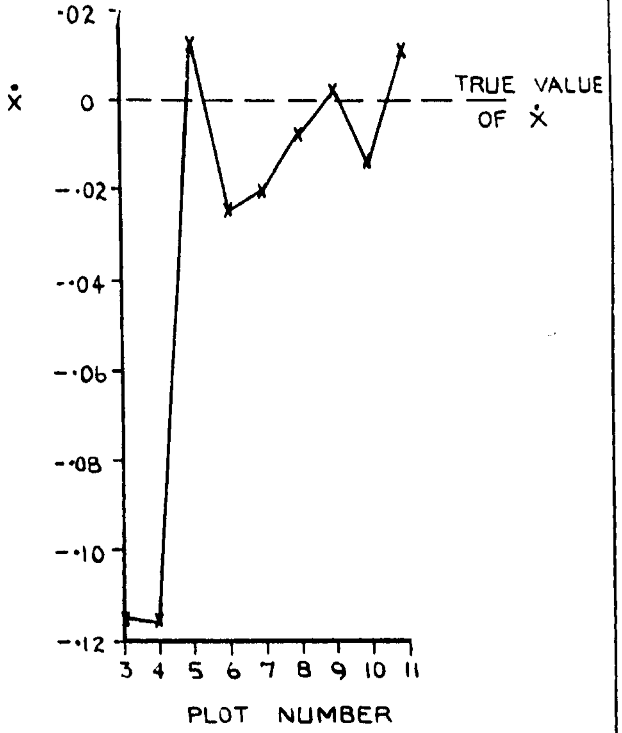
SECOND ORDER POLAR KALMAN FILTER

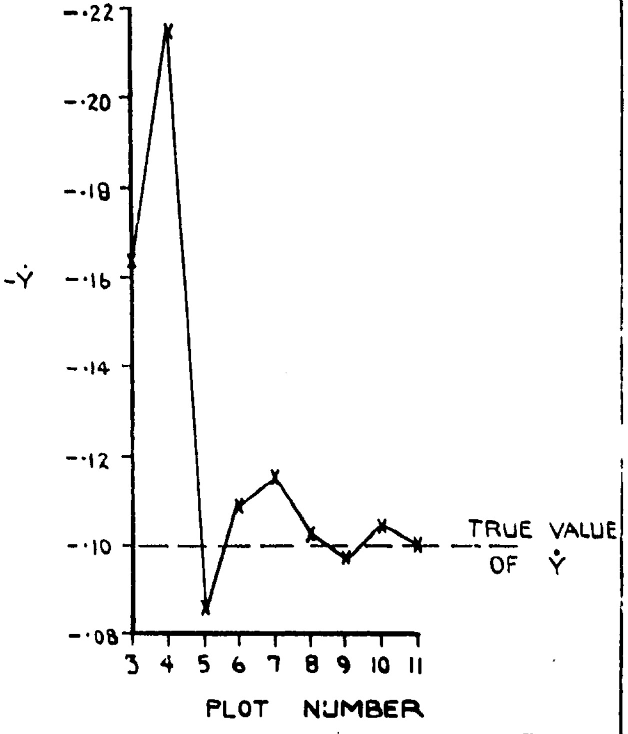
3

---●--- INPUT POSITIONS
---x--- SMOOTHED POSITIONS

CROSSING DISTANCE =10
DATA INTERVAL = 5 TIME UNITS



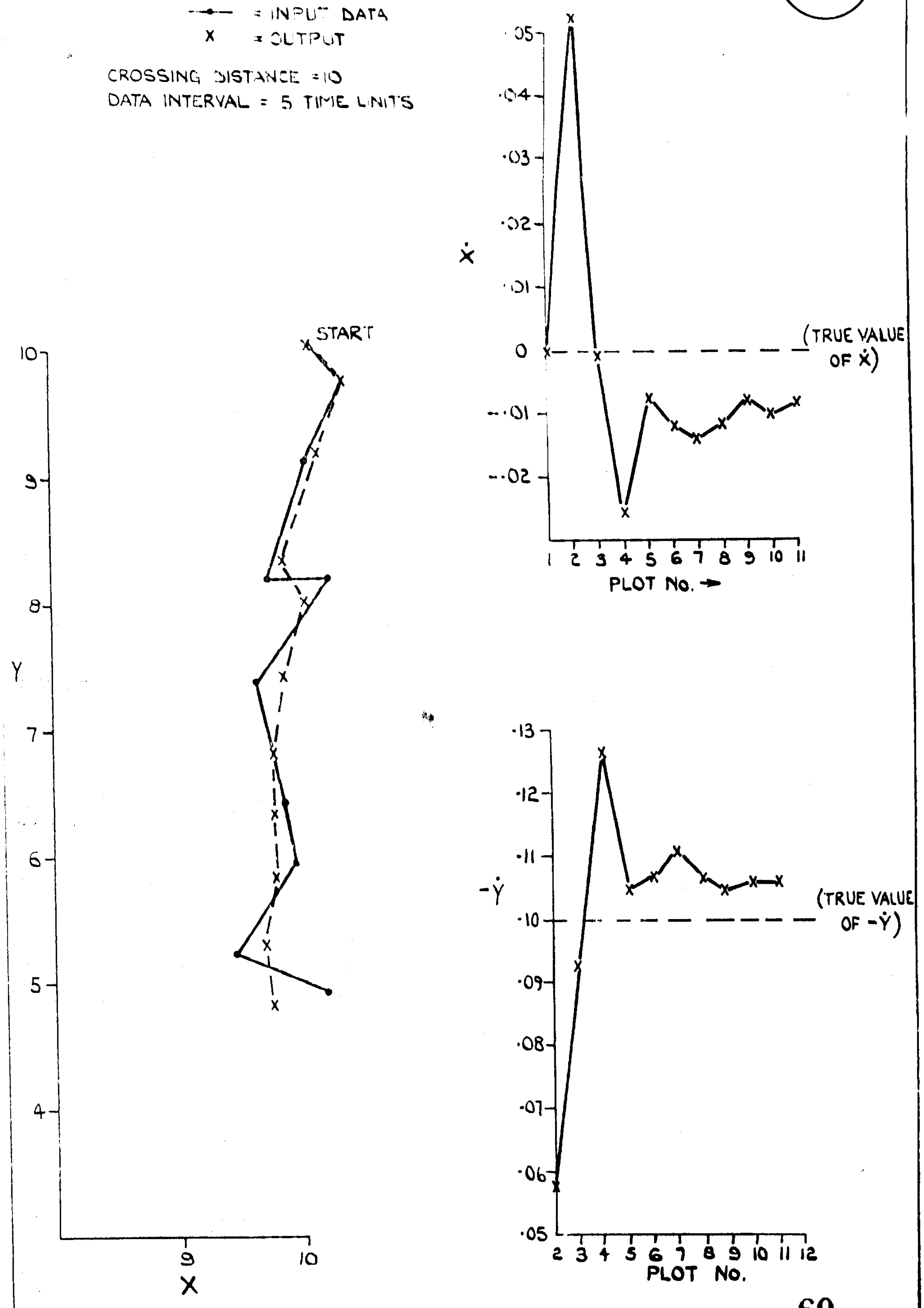




A.S.W.E. DATE. 19.4.72. TR. C. HARRISON CH. APR. DRG. No.

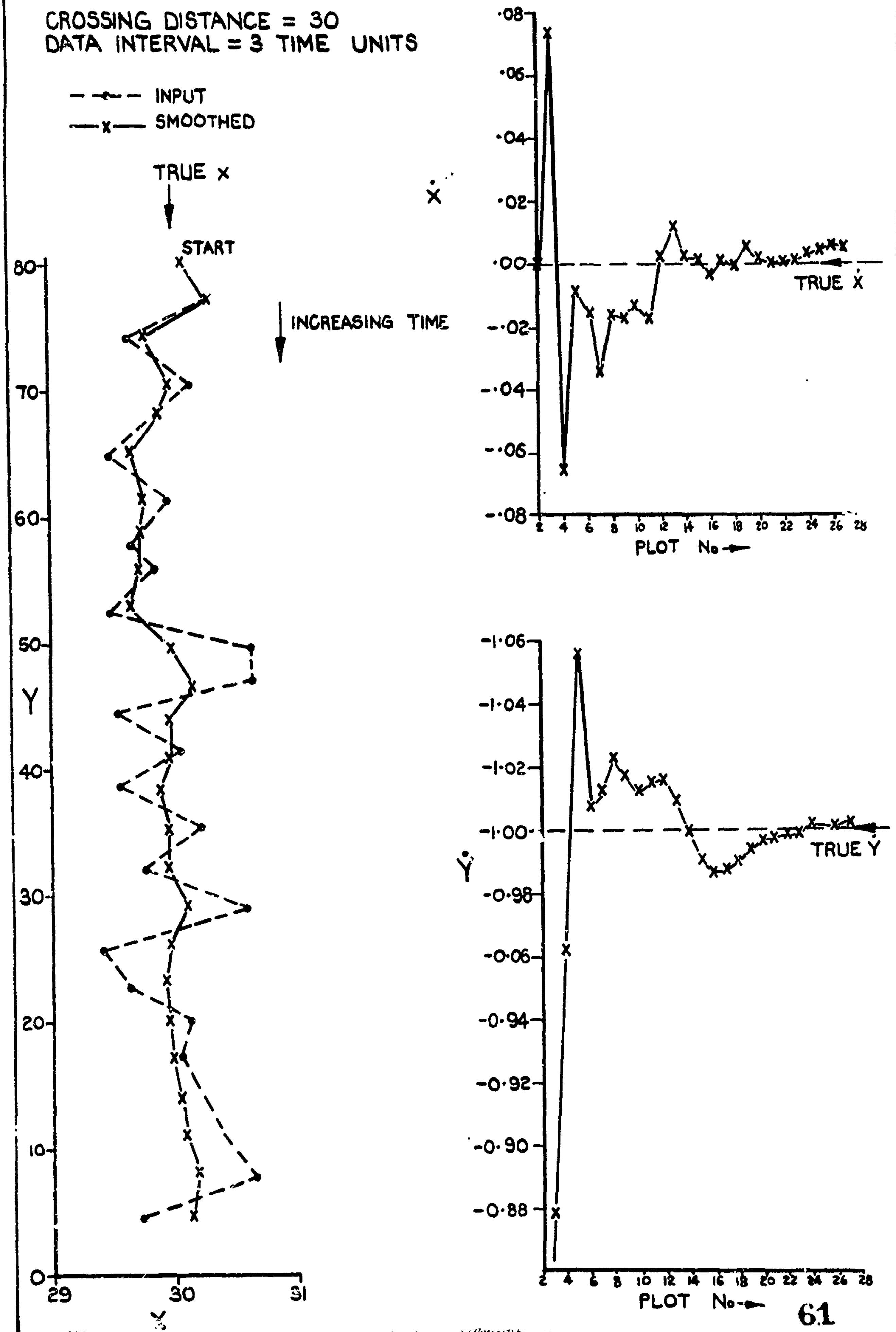
XY KALMAN TRACKER (FIRST ORDER)

4

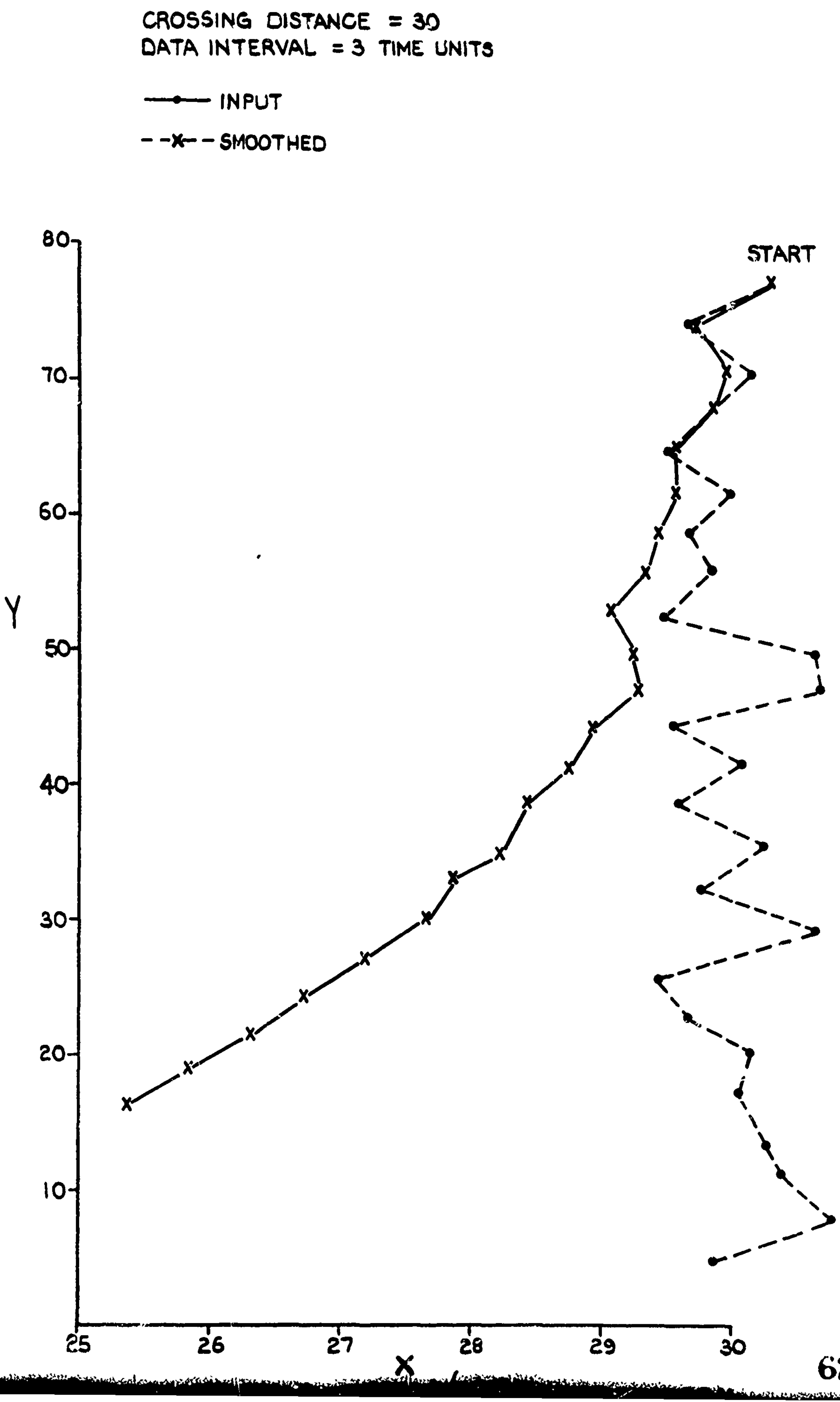


A.S.W.E DATE 18.4.72 TR [illegible] DRG No.

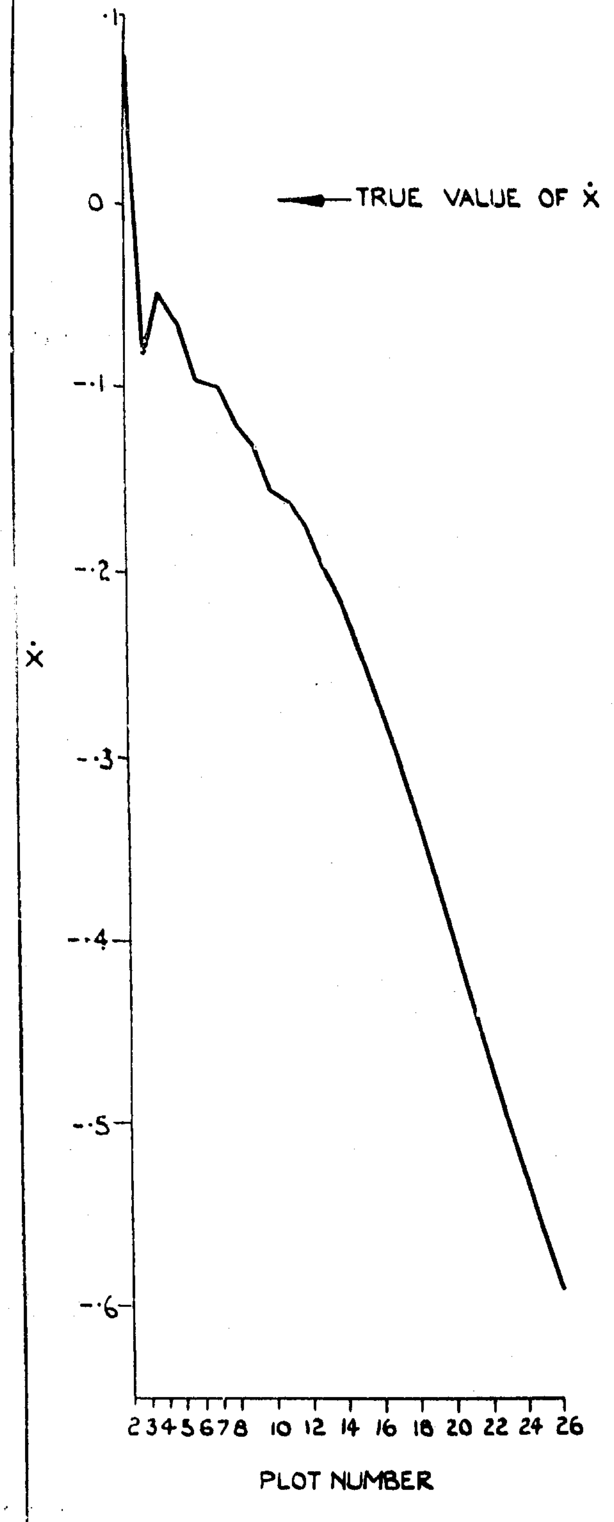
FIRST ORDER CARTESIAN TRACKER

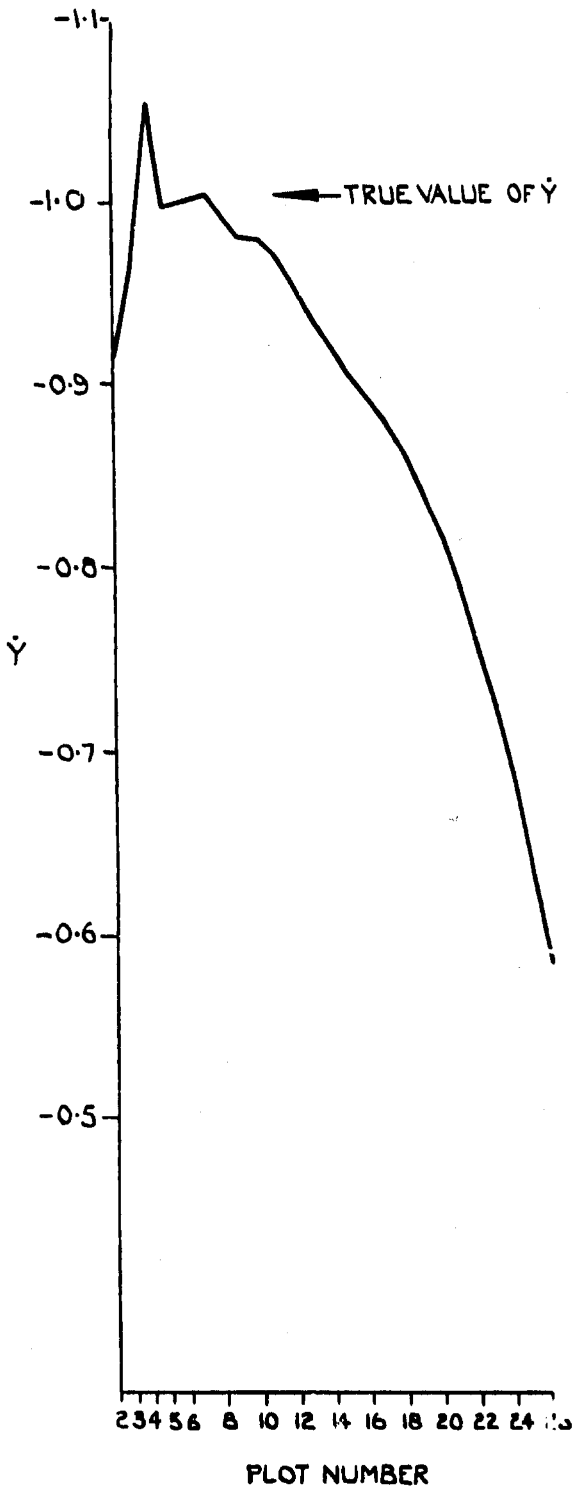


FIRST ORDER POLAR KALMAN



FIRST ORDER Rθ KALMAN

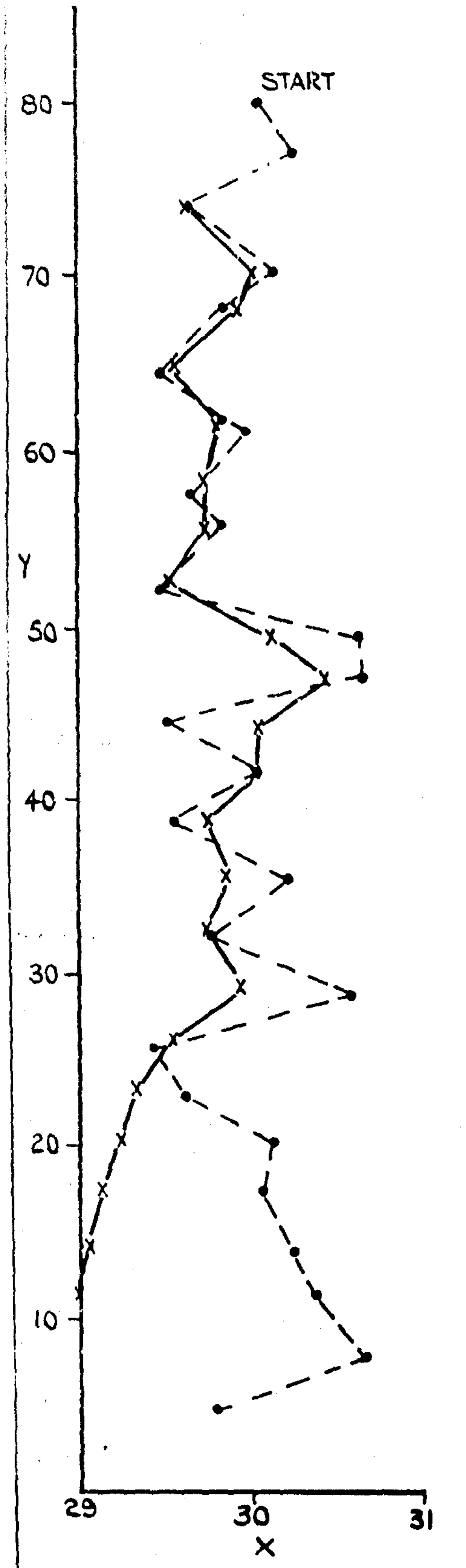


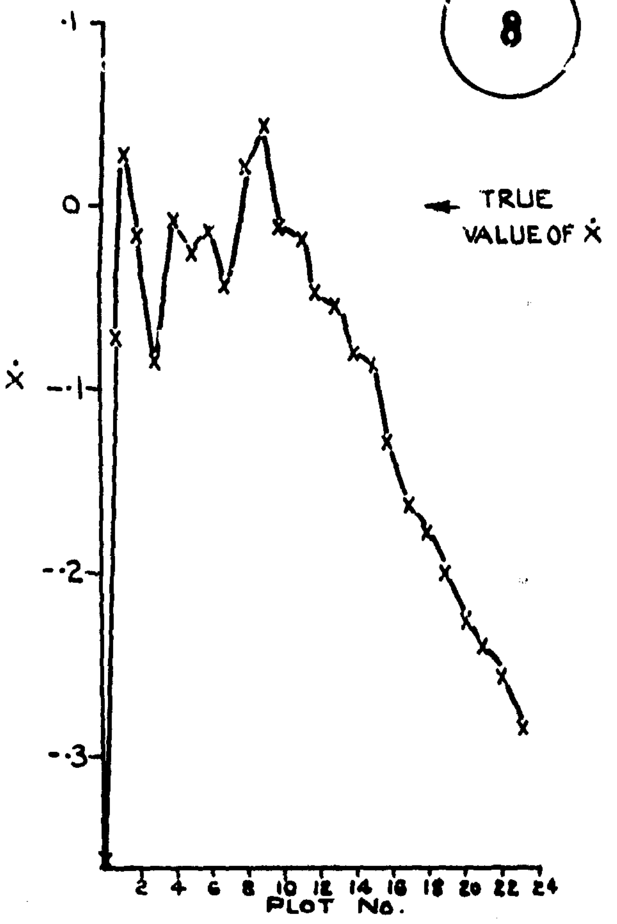


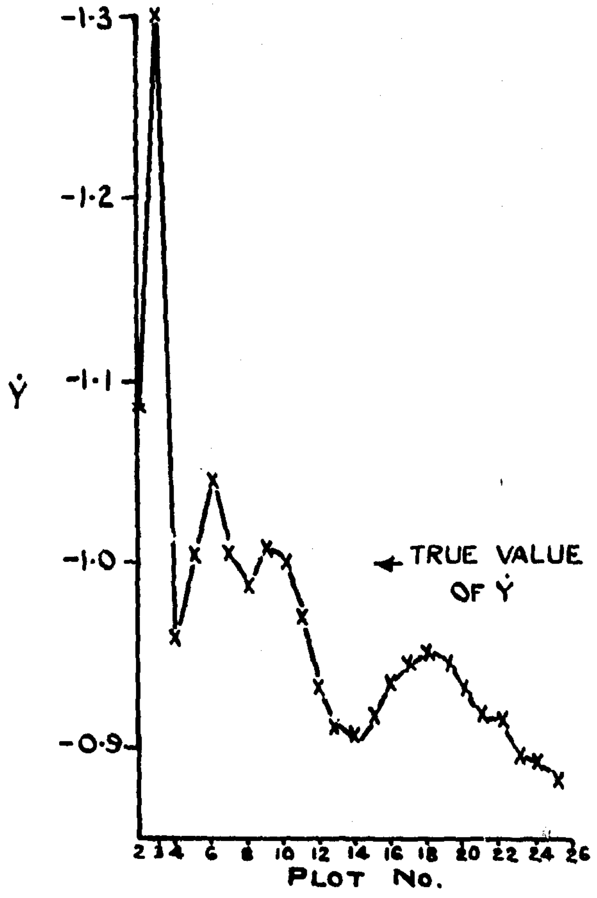
SAME CONDITIONS AS FOR FIGURE 6

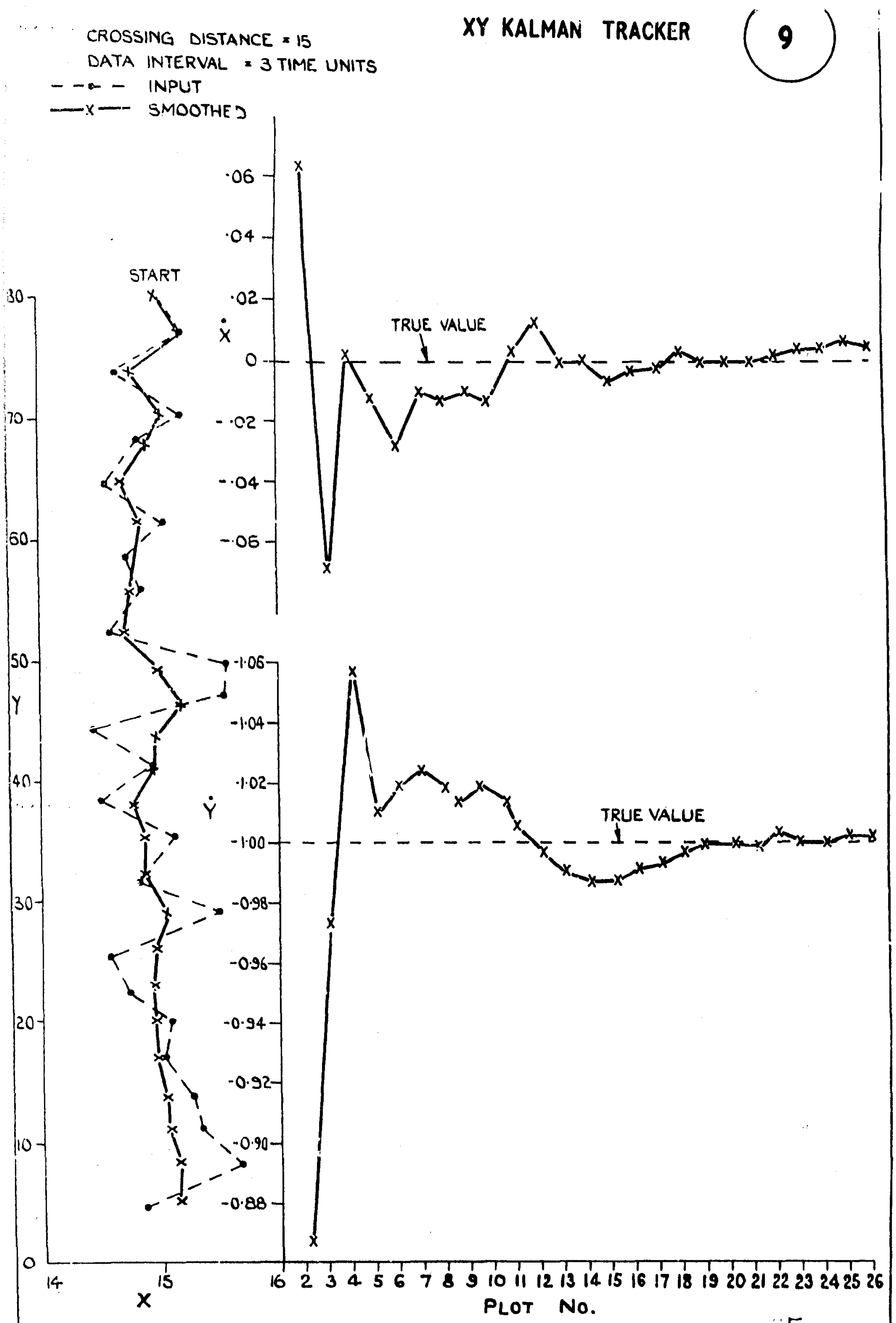
A.S.W.E. DATE 18·4·72 TR. [illegible] DRG No.

ZND ORDER Rθ KALMAN TRACKER



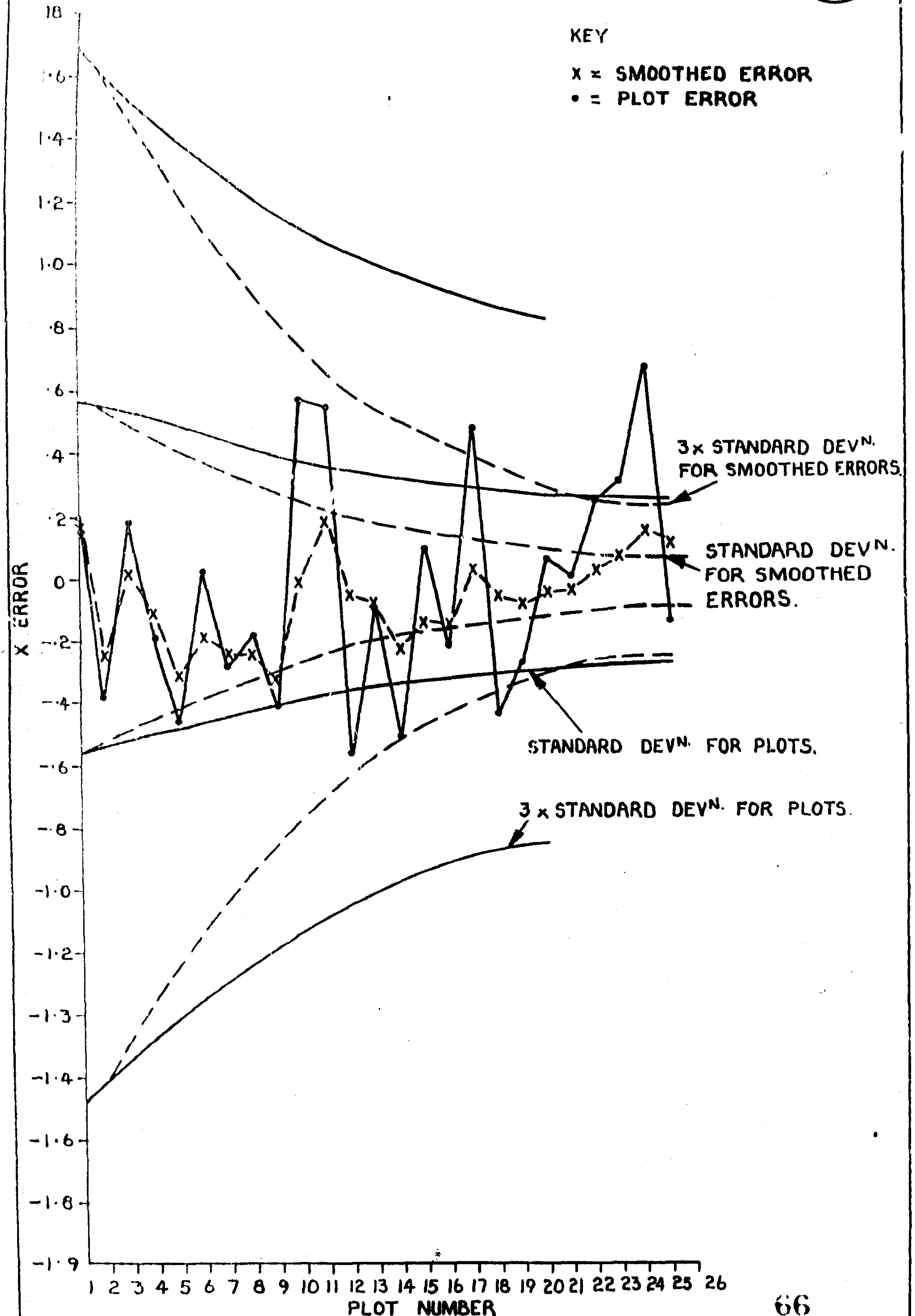






DATE ... TR ... DRG.No.

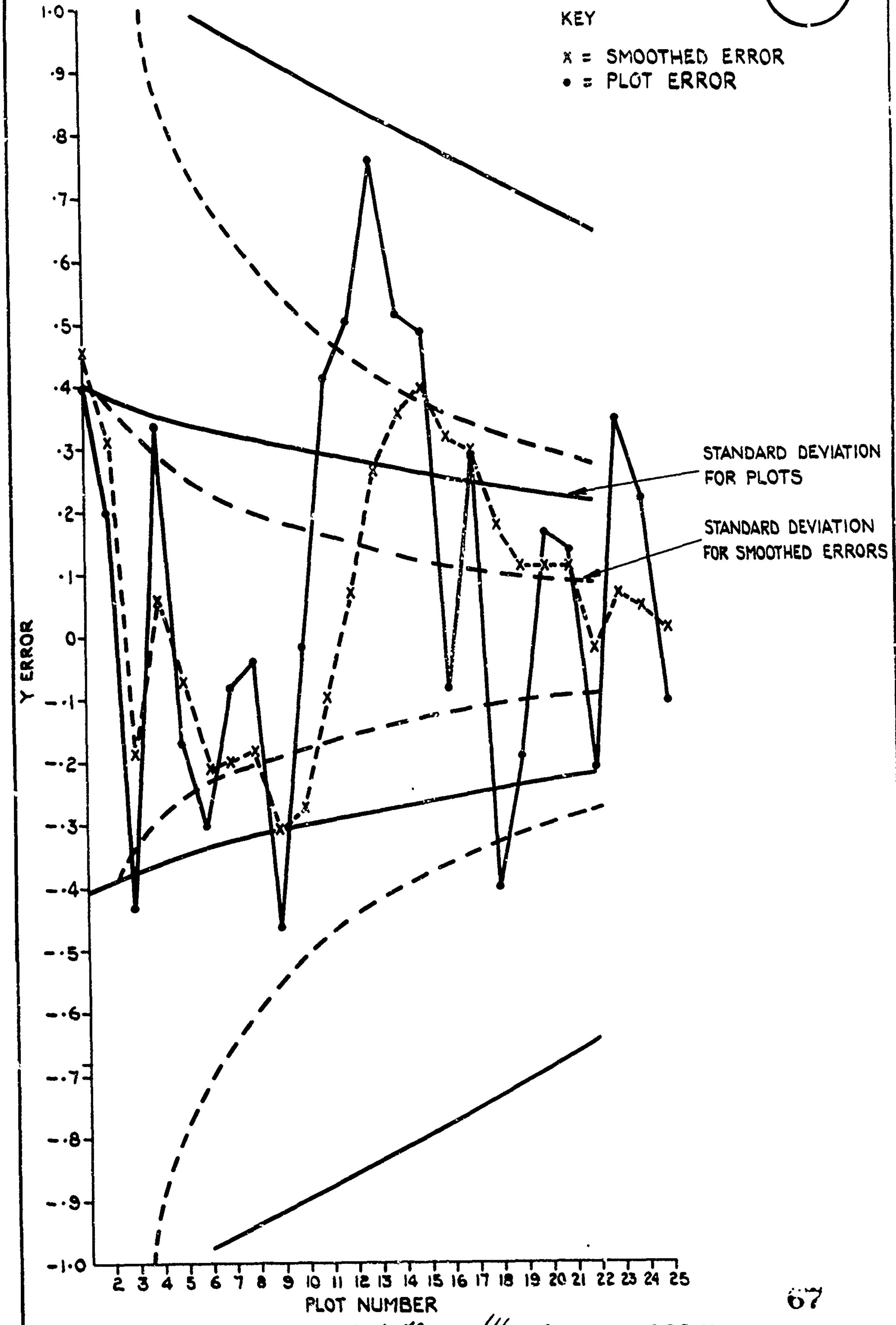
X-ERRORS FROM FIG. 9 AND THEIR A PRIORI STANDARD DEVNS. (10)



A.S.W.E. DATE 17.4.72 TR. C. Godfrey. DRG. No.

Y-ERRORS FROM FIG. 9 AND THEIR A PRIORI STANDARD DEVIATIONS

11

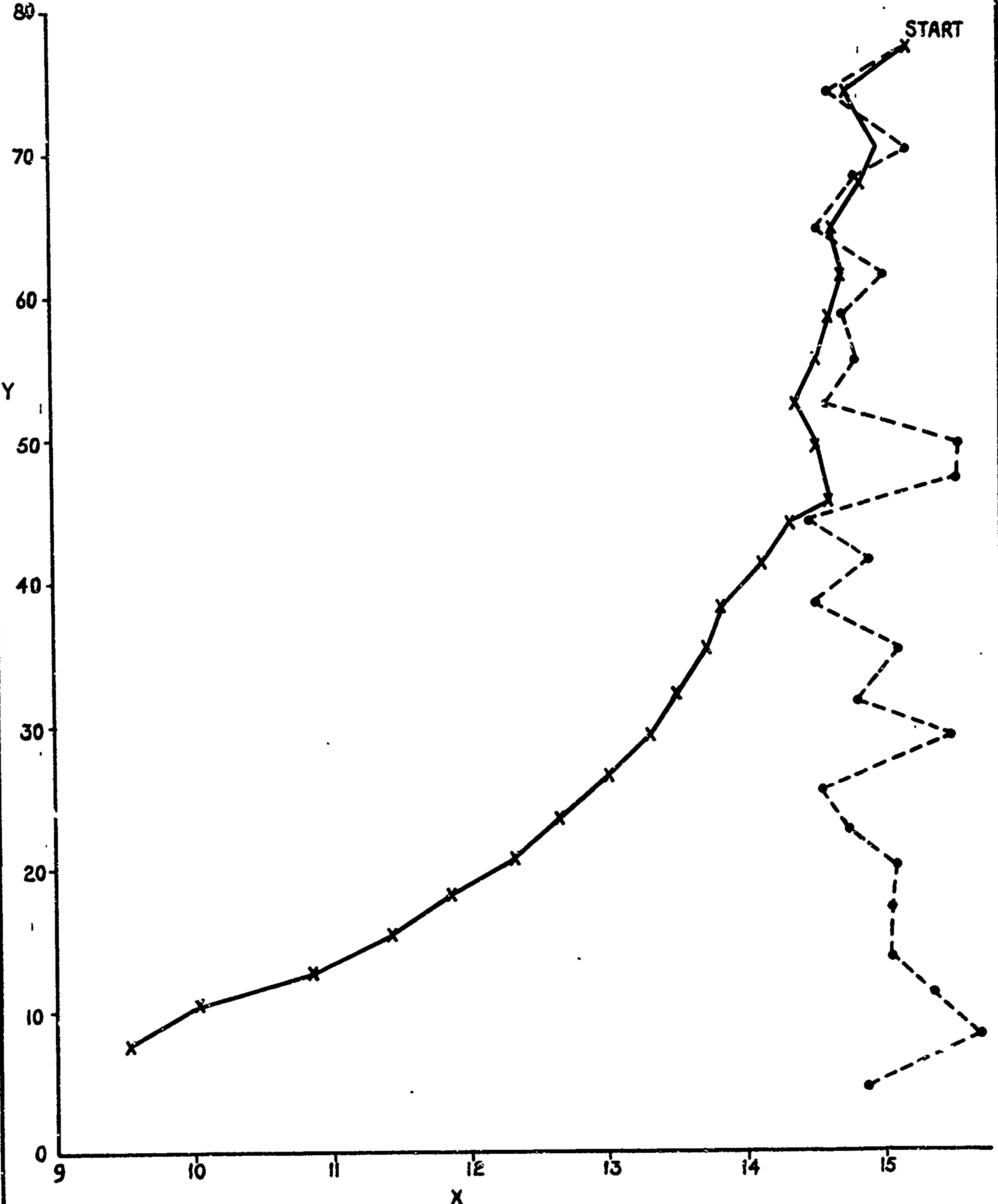


A.S.W.E. DATE 18-4-72 TR. [illegible] DRG. No.

1ST ORDER Rθ KALMAN TRACKER

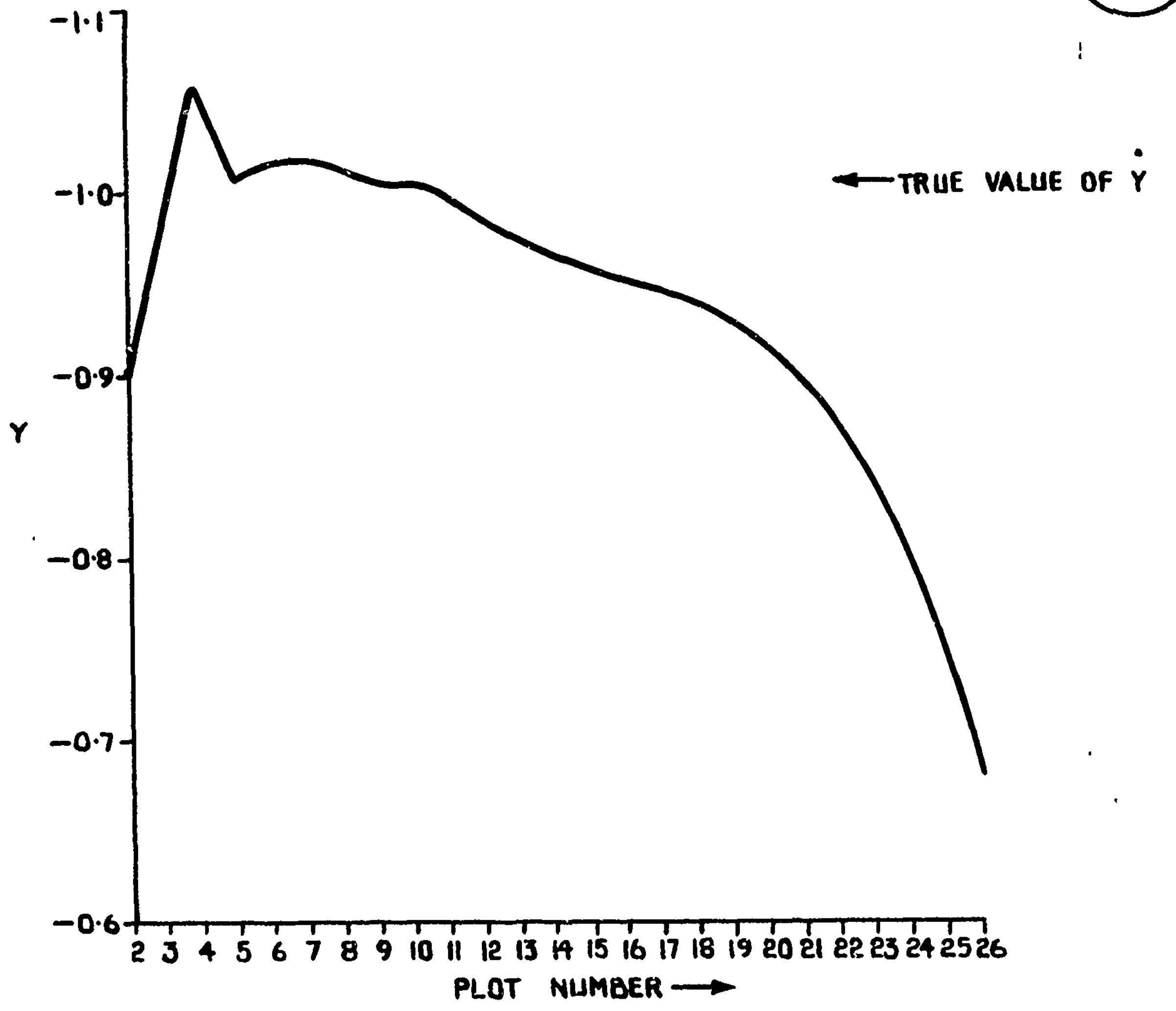
---●--- INPUT
—X— SMOOTHED

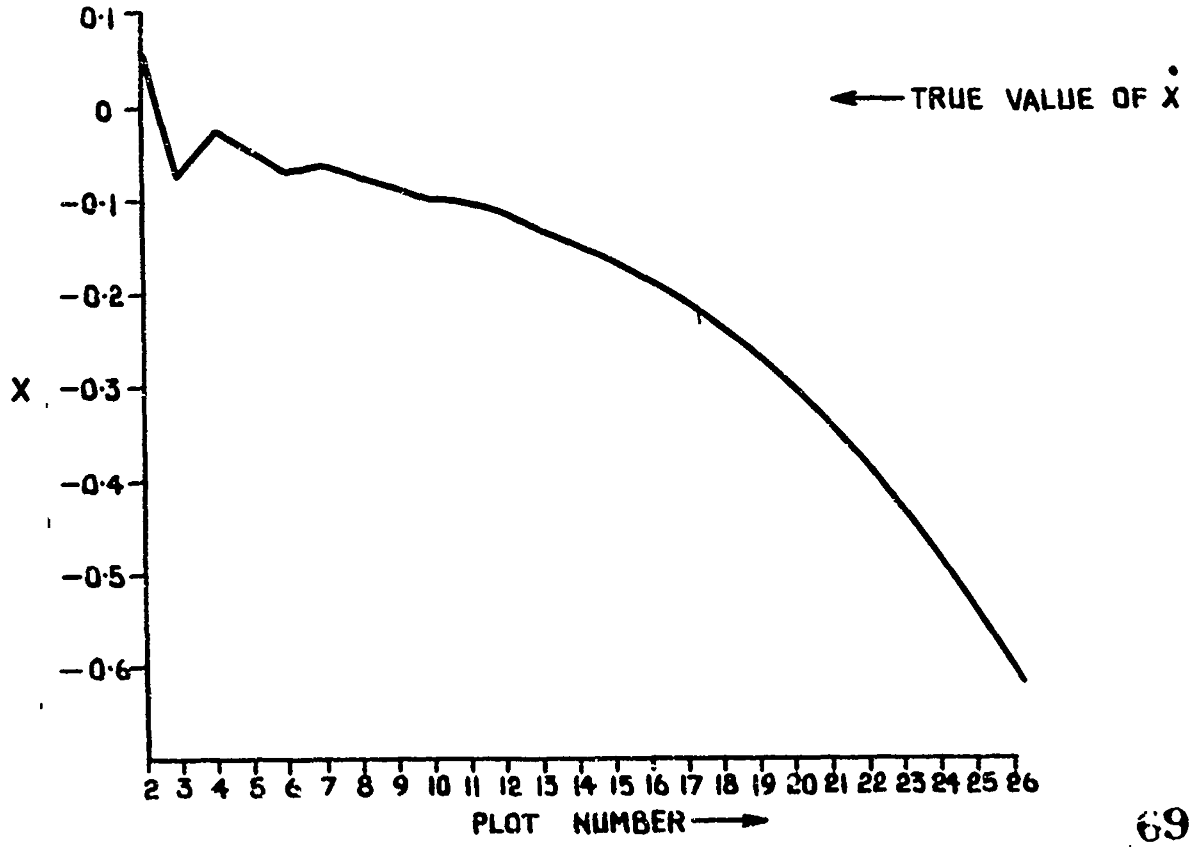
CROSSING DISTANCE = 15
DATA INTERVAL = 3 TIME UNITS



A.S.W.E. DATE 24-4-72 TR [illegible] CH APP DRG No 68

FIRST ORDER Rθ KALMAN TRACKER

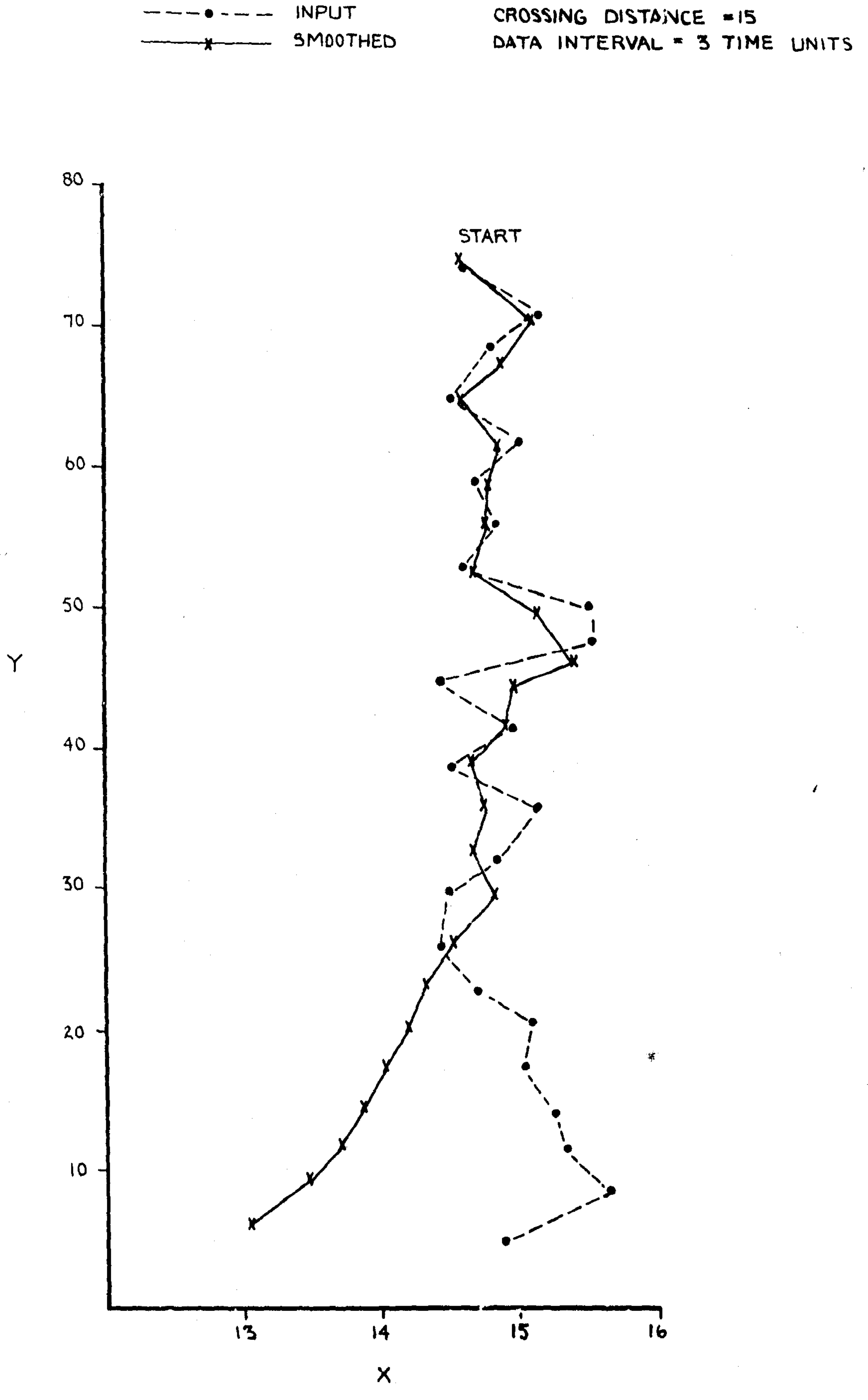




A.S.W.E. DATE 19-4-78 TR. C. Stoud CH. APR DRG. No.

SECOND ORDER Rθ KALMAN TRACKER

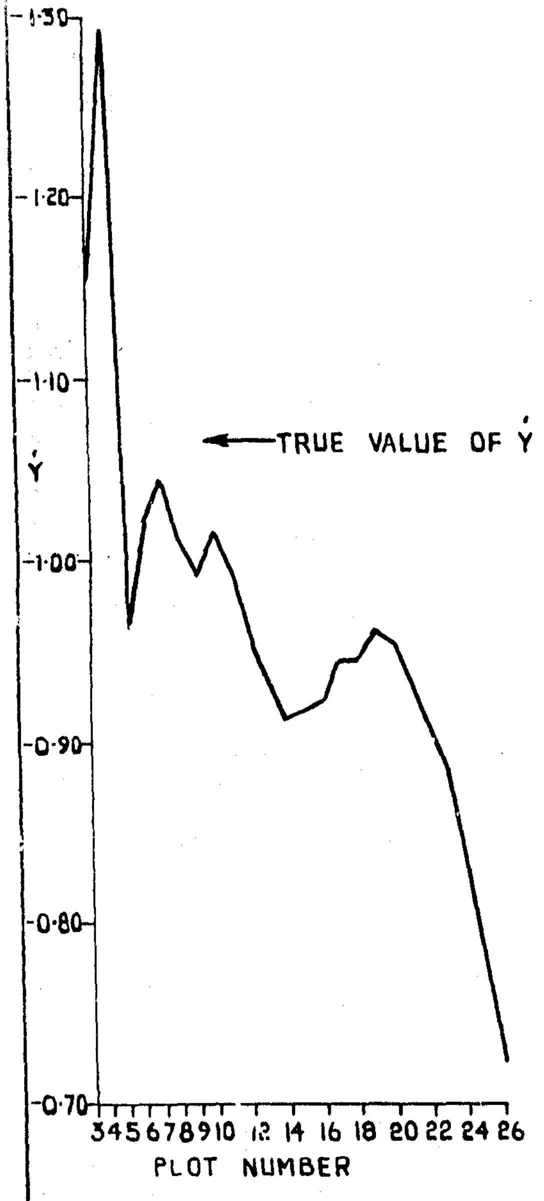
14

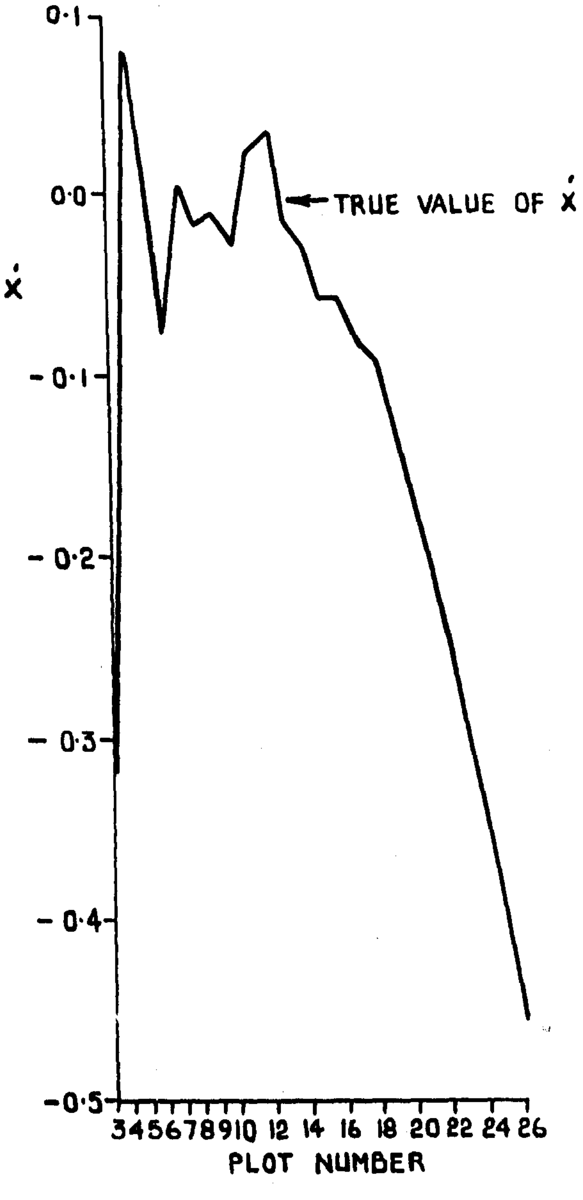


A.S.W.E. DATE. 18.4.72. TR. C. Harrison CH. APR. DRG. No

SECOND ORDER Rθ KALMAN TRACKER

15





71

A.S.W.E. DATE 20.4.72. TR. C. Stroud. CH. APP. DRG No